THE KOREA INDUSTRIAL PROPERTY OFFICE

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Industrial Property Office.

Application Number : Patent Application No. 00-70631

Application Date : 25 November 2000

Applicant : Pohang University of Science and Technology Foundation

27 December 2000

COMMISSIONER

1020000070631 2000/12/2

[Document]	Patent Application
[Right]	Patent
[Receiver]	Commissioner
[Reference No.]	1
[Filing Date]	2000.11.25
[Classified No.]	G06F
[Title]	Apparatus and method for digital multiplication using redundant binary arithmetic
[Applicant]	
[Name]	Pohang University of Science and Technology Foundation
[Applicant's code]	2-1999-900096-8
[Attorney]	
Name:	Young-pil Lee
Attorney's code:	9-1998-000334-6
General Power of Attorney	
Registration No.:	1999-050323-2
[Attorney]	
Name:	Heung-Soo Choi
Attorney's code:	9-1998-000657-4
General Power of Attorney	
Registration No.:	1999-050350-5
[Attorney]	
Name:	Hae-Young Lee
Attorney's code:	9-1999-000227-4
General Power of Attorney	
Registration No.:	2000-006267-7
[Inventor]	
[Name]	PARK, Hong Joon
[Number]	561011-1902227
[Zip Code]	790-390

[Address]	9-802 Kyosu Apt., Jigok-dong, Nam-gu, Pohang-city Kyungsangbuk-do, Seoul, Republic of Korea
[Nationality]	Republic of Korea
[Inventor]	
[Name]	LEE, Sang-Hoon
[Number]	720131-1335010
[Zip Code]	790-390
[Address]	19-219 Gisuksa, Jigok-dong, Nam-gu, Pohang-city Kyungsangbuk-do, Seoul, Republic of Korea
[Nationality]	Republic of Korea
[Examination Request]	Requested

1020000070631 2000/12/2

[Application Order] I/We file as above according to Art. 42 of the Patent Application and request examination according to Art. 60 of the Patent Application.

Attorney	Young-pil Lee
Attorney	Heung-Soo Choi
Attorney	Hae-Young Lee

[Fee]

Basic fee:	20 Sheet(s)	29,000 won
Additional fee:	31 Sheet(s)	31,000 won
Priority claiming fee:	0 Case(s)	0 won
Examination fee:	12 Claim(s)	493,000 won
Total fee:	553,000 won	
Reason for fee reduction	Educational organization	
Fee after reduction	276,500 won	

[Enclosures] 1. Abstract and Specification (and Drawings)-1 copy

대 한 민 국 특 허 청
KOREAN INDUSTRIAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Industrial Property Office.

출 원 번 호 : 특허출원 2000년 제 70631 호
Application Number

출 원 년 월 일 : 2000년 11월 25일
Date of Application

출 원 인 : 학교법인 포항공과대학교
Applicant(s)

2000 년 12 월 27 일

특 허 청
COMMISSIONER

【서류명】 특허출원서
【권리구분】 특허
【수신처】 특허청장
【참조번호】 0001
【제출일자】 2000.11.25
【국제특허분류】 G06F
【발명의 명칭】 잉여 이진수 연산을 채택한 디지털 곱셈 장치 및 방법
【발명의 영문명칭】 Apparatus and method for digital multiplication using redundant binary arithmetic
【출원인】
【명칭】 학교법인 포항공과대학교
【출원인코드】 2-1999-900096-8
【대리인】
【성명】 이영필
【대리인코드】 9-1998-000334-6
【포괄위임등록번호】 1999-050323-2
【대리인】
【성명】 최흥수
【대리인코드】 9-1998-000657-4
【포괄위임등록번호】 1999-050350-5
【대리인】
【성명】 이해영
【대리인코드】 9-1999-000227-4
【포괄위임등록번호】 2000-006267-7
【발명자】
【성명의 국문표기】 박홍준
【성명의 영문표기】 PARK,Hong Joon
【주민등록번호】 561011-1902227
【우편번호】 790-390
【주소】 경상북도 포항시 남구 지곡동 교수숙소 9동 802호
【국적】 KR
【발명자】
【성명의 국문표기】 이상훈
【성명의 영문표기】 LEE,Sang Hoon

【주민등록번호】 720131-1335010

【우편번호】 790-390

【주소】 경상북도 포항시 남구 지곡동 756번지 남자기숙사 19동 219호

【국적】 KR

【심사청구】 청구

【취지】 특허법 제42조의 규정에 의한 출원, 특허법 제60조의 규정에 의한 출원심사 를 청구합니다. 대리인
이영필 (인) 대리인
최흥수 (인) 대리인
이해영 (인)

【수수료】

【기본출원료】	20	면	29,000 원
【가산출원료】	31	면	31,000 원
【우선권주장료】	0	건	0 원
【심사청구료】	12	항	493,000 원

【합계】 553,000 원

【감면사유】 학교

【감면후 수수료】 276,500 원

【첨부서류】 1. 요약서·명세서(도면)_1통

【요약서】

【요약】

잉여 이진수 연산을 채택한 디지털 곱셈 장치 및 방법이 개시된다. 2^k진수 수 체계를 사용하여 두 수(X 및 Y)를 곱셈하는 이 장치는, m비트의 Y를 m/k 디지트의 $D_{m/k-1}D_{m/k-2} \ldots D_i \ldots D_1D_0$)로 데이타 변환하는 데이타 변환부와, 데이타 변환부에서 변환된 Y의 각 디지트 Di를 기본 배수의 계수들의 조합으로 변환하고, 변환된 조합을 X와 승산하고, 승산된 결과를 잉여 이진수 형태의 부분곱으로서 출력하는 부분곱 산출부와, 변환된 Y의 모든 디지트들에 대한 부분곱들을 가산하는 잉여 이진수 가산부와, 잉여 이진수 형태의 가산된 결과를 일반 이진수 형태로 변환하고, 변환된 일반 이진수 형태의 결과를 두 수들의 곱셈 결과로서 출력하는 RB-NB 변환부를 구비하는 것을 특징으로 한다. 그러므로, 진수를 확장시키면서도 하드웨어 부담을 최소화시킬 수 있으며, 곱셈기가 중요 구성 기능 블록인 여러 시스템들이 좀 더 간단하게 구현될 수 있도록 하는 효과가 있다.

【대표도】

도 2

【명세서】

【발명의 명칭】

잉여 이진수 연산을 채택한 디지털 곱셈 장치 및 방법{Apparatus and method for digital multiplication using a redundant binary arithmetic}

【도면의 간단한 설명】

도 1은 본 발명에 의한 잉여 이진수 연산을 채택한 디지털 곱셈 장치의 개략적인 블럭도이다.

도 2는 도 1에 도시된 부분곱 산출부의 본 발명에 의한 바람직한 실시예의 블럭도이다.

도 3은 도 2에 도시된 멀티플렉서들중 하나 및 그에 대응하는 논리 조합부의 본 발명에 의한 바람직한 일실시예의 회로도이다.

도 4는 간략화된 가산기의 본 발명에 의한 바람직한 일실시예의 회로도이다.

도 5는 본 발명에 의한 가산기의 바람직한 다른 실시예의 회로도이다.

도 6은 본 발명에 의한 디지털 곱셈 방법을 설명하기 위한 플로우차트이다.

도 7은 도 6에 도시된 기본 배수의 계수들을 결정하는 단계를 설명하기 위한 본 발명에 의한 플로우차트이다.

도 8은 도 6에 도시된 제152 단계의 본 발명에 의한 바람직한 일실시예의 플로우차트이다.

도 9는 도 6에 도시된 제152 ~ 제156 단계들을 수행하는 본 발명에 의한 잉여 이진수 연산을 채택한 디지털 곱셈 방법의 바람직한 일실시예의 플로우차트이다.

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<10> 본 발명은 병렬 곱셈기에 관한 것으로서, 특히, 부분곱 생성을 위해 잉여 이진수 연산 기법을 채택한 디지털 곱셈 장치 및 방법에 관한 것이다.

<11> 최근들어서 시스템-온-칩(SoC:system-on-a-chip)의 경향에 따라, 집적회로를 구성하는 각 기능 블록들은 향상된 성능을 가지면서도 적은 하드웨어 양을 차지할 것이 요구되고 있다. 이러한 기능 블록들중 가장 대표적인 것으로서 곱셈기가 있으며, 곱셈기는 여러 고성능 마이크로 프로세서나 신호 처리 칩에서 중요한 역할을 한다. 따라서, SoC의 경향에 부합될 수 있는 하나의 방법으로서, 곱셈기의 성능을 개선하면서도 크기를 줄이는 것이 요구되고 있다.

<12> 종래의 곱셈기에서 사용되는 곱셈 연산 알고리즘을 살펴보면, 부스(Booth)에 의한 방식을 변형시킨 변형 부스 알고리즘(MBA:Modified Booth's Algorithm)을 이용해서 부분곱을 만든 후, 왈라스 트리(wallace-tree)와 같은 구조의 캐리 저장(carry-save) 덧셈기를 이용해서 부분곱들을 더하여 최종 곱셈 결과를 구하는데 대개 일반 이진 연산(normal binary arithmetic)으로 곱셈이 이루어진다. 여기서, 부분곱들을 더하는 부분에서 일반 이진 연산 외에도 잉여 이진 연산 (redundant binary arithmetic) 방식을 적용하기도 한다. 이 때, 잉여 이진 연산의 특징인 연속된 캐리(carry)의 전파가 없다는 것은 일반적으로 부분곱들을 더하는 연산에서 요구되는 덧셈기의 성질이다.

<13> 잉여 이진 연산을 이용한 곱셈기의 종래의 연산 방법들중의 하나를 다음과 같이 살

펴본다.

<14> 종래의 연산 방법에 의하면, 먼저 일반 이진 연산으로 곱셈 입력값에 따른 부분곱을 MBA를 이용해서 만들어낸다.

<15> 전술한 MBA를 이용한 종래의 곱셈 방법은 4 진수(radix-4)의 수 체계를 사용한다. 즉, 두 수들 X와 Y의 곱셈 연산을 할 때 MBA를 이용해서 m비트인 Y를 다음수학식 1과 같이 나타낸다.

<16> 【수학식 1】

$$Y = \sum_{i=0}^{\frac{m}{2}-1} D_i \cdot 4^i$$

<17> 여기서 Di는 {-2,-1,0,1,2}의 값들중 하나에 해당하고, Y의 연속된 세 비트들(y_{i+2}, y_{i+1}, y_i)을 $-2 \cdot y_{i+2} + y_{i+1} + y_i$의 계산에 의해서 부호화(recoding)함으로서 구해진다. 이 연속된 세 비트들을 m비트인 Y에서 한 비트씩 겹치면서 묶으므로, 실제적으로 $\lceil \frac{m}{2} \rceil$ 개의 3 비트 묶음이 나오게 되며 부분곱의 수도 m에서 $\lceil \frac{m}{2}+1 \rceil$로 감소하게 되는 이점이 있다. 여기서, $\lceil x \rceil$는 x보다 큰 가장 작은 정수를 의미한다. 이 때, +1이 추가된 이유는 2의 보수 연산에 의해서 각 부분곱의 최하위 유효 비트에 더해 주어야 할 보정 비트들을 모아서 하나의 부분곱으로 고려했기 때문이다. m비트인 Y를 MBA 방식에서 부호화하기 위해서 3 비트씩 묶는 방법은 다음 수학식 2와 같다.

<18> 【수학식 2】

$$\underbrace{y_{m-1}}_{D_{\frac{m}{2}-1}} \cdots \overbrace{\underbrace{y_{i-1}y_{i-2}}_{D_{\frac{i}{2}}}\underbrace{y_{i-3}}_{D_{\frac{i}{2}-2}}}^{} \cdots \overbrace{\underbrace{y_2y_1}_{D_1}\underbrace{y_0\ 0}_{D_0}}^{}$$

<19> 전술한 수학식 2는 m이 짝수인 경우에 적용되는 방법이며, 항상 최하위 유효 비트

아래에 '0'을 덧붙인다. MBA는 4진수로 부호화하므로 한 비트씩 겹치면서 3 비트씩 묶었으며, 일반적으로 2^k진수로 부호화하는 경우 한 비트씩 겹치면서 k+1비트씩 묶는다. 이 때, Di에 X를 승산하여 일반 이진수로 된 부분곱들을 만든다.

<20> 다음에, 일반 이진수로된 부분곱들을 잉여 이진수(redundant binary number)로 표현된 부분곱들로 변환한다. 즉, 두개의 일반 이진수 형태를 갖는 부분곱들 A 및 B의 합은 다음 수학식 3과 같이 표현될 수 있다.

<21> 【수학식 3】

$$A+B=A-(-B)=A-(\overline{B}+1)=(A-\overline{B})-1$$

<22> 여기서, a-b≡(a,b)라고 정의하면, 다음 수학식 4로부터 -1≡(0,1)이 됨을 알 수 있다.

<23> 【수학식 4】

$$A-\overline{B}\equiv(A,\overline{B})$$

<24> 여기서, 수학식 4의 우변은 A+B 연산 결과의 잉여 이진수 표현에 해당한다. 수학식 3의 우변에서 최종적으로 최하위 유효 비트에 가산되는 -1은 2의 보수 연산에 의해서 더해야 하는 보정 비트이다. 이 때, -1은 잉여 이진수로 (0,1)로 표현 할 수 있다. 수학식 3을 일반 이진수 형태로 된 부분곱들을 덧셈할 때 적용하면, 여러 개의 일반 이진수 부분곱들을 두 개씩 짝을 지운 후 한 개를 비트 별로 반전 시키면 잉여 이진수로 표현된 부분곱의 합을 구할 수가 있다.

<25> 결국, MBA에 의할 경우, m비트의 곱셈 입력 값에 대한 곱셈일 때 $\lceil \frac{m}{2} \rceil + 1$개의 일반 이진수 부분곱들이 만들어 진다. 따라서, r개의 일반 이진수 부분곱들을 잉여 이진수

로 변환하면서 한 번 더하면, $\lceil \frac{r}{2} \rceil$ 개의 잉여 이진수로 표현된 부분곱들의 합이 만들어지게 된다.

<26> 이러한 잉여 이진수들을 다음 표 1과 같은 덧셈 규칙에 의한 잉여 이진 연산에 의해 더하여 최종적으로 X와 Y의 잉여 이진수 형태의 곱셈 결과가 얻어지게 된다.

<27> 표 1은 종래의 연산 방식에 의해서 잉여 이진수(redundant binary number) 덧셈을 하는 규칙을 나타낸다.

<28> 표 1에서, (a_i^+, a_i^-) 및 (b_i^+, b_i^-)는 더하고자 하는 두개의 잉여 이진수들을 의미하고, (c_i^+, c_i^-) 및 (s_i^+, s_i^-)는 각각 두 수들을 더했을 때의 중간 캐리 및 중간 합을 잉여 이진수로 각각 표현한 것에 해당하고, any는 '0' 또는 '1'의 값중 어느 것이라도 상관없다는 의미이다. 또한, h_{i-1}은 연속되는 캐리의 전파를 막기 위해서 만들어진 중간 변수로서, h_i가 1 인 경우의 (a_i^+, a_i^-) 및 (b_i^+, b_i^-)의 값들은 굵은 글씨로된 부분에 해당한다. h_i 가1 인 경우는 -1 값이 캐리로서 전파될 가능성이 있는 입력 패턴에 해당한다.

<29> 【표 1】

CASE	(a_i^+, a_i^-) (b_i^+, b_i^-)	h_{i-1}	(c_i^+, c_i^-) (s_i^+, s_i^-)
1	(0,0) (1,0) (0,1)	any	(0,0) (0,0)
2	(0,1) (0,0)	0	(0,0) (0,1)
	(0,0) (0,1)	1	(0,1) (1,0)
3	(1,0) (0,0)	0	(1,0) (0,1)
	(0,0) (1,0)	1	(0,0) (1,0)
4	(0,1)	any	(0,1) (0,0)
5	(1,0)	any	(1,0) (0,0)

<30> 표 1을 참조하면, 두개의 잉여 이진수들[(a_i^+,a_i^-) 및 (b_i^+,b_i^-)]의 덧셈은 다음과 같이 5 가지의 경우들중 하나에 해당한다.

<31> Case 1은 덧셈 결과가 0인 경우로서, 중간 캐리와 중간 합이 모두 0이다. Case 2는 덧셈 결과가 -1인 경우로서, 만일 -1이 아래 자리로부터 넘어올 가능성이 있으면(즉, h_i가 1 이면) 중간 합은 1이고 중간 캐리는 -1(실제로는 한 자리수가 넘어가므로 -2에 해당)이 되어서 -1이 넘어 오더라도 중간 합인 1과 상쇄되어 더 이상의 캐리 전파가 없도록 한다. Case 3은 덧셈 결과가 1인 경우로서, 만일 1이 아래 자리에서 넘어올 가능성이 있으면(즉, h_i가 0 이면) 중간 합은 -1이고 중간 캐리는 1(실제로는 한 자리수가 넘어가므로 2에 해당)이 되어서 1이 넘어 오더라도 중간 합인 -1과 상쇄돼서 더 이상의 캐리 전파가 없도록 하고 h_i가 1이면 아래 자리에서 -1이 넘어 올 가능성이 있으므로 중간 합은 1로 하고 캐리를 0으로 해서 -1이 넘어 오더라도 중간 합 1과 상쇄돼서 더 이상의 캐리 전파가 없게 된다. Case 4 및 case 5들은 덧셈 결과가 각각 -2 및 2인 경우로서, 아래 자리에서 넘어오는 캐리의 값에 상관 없이 더 이상의 캐리 전파는 없고 중간 합은 0이고 캐리는 각각 -1 과 1이다. 표 1에 의해 두 잉여 이진수를 더할 경우 잉여 이진수들[(a_i^+,a_i^-) 또는 (b_i^+,b_i^-)]의 값이 (1,1)의 값을 갖는 경우는 제외되었다. 따라서 잉여 이진수 형태의 부분곱을 만들 때, (1,1)형태의 잉여 이진수는 (0,0)으로 만들어야 한다. 표 1에 의해서 두 잉여 이진수 덧셈 결과인 (d_i^+,d_i^-)는 다음 수학식 5와 같이 표현될 수 있다.

<32> 【수학식 5】

$$d_i^+=(s_i^++c_{i-1}^+)\cdot\overline{(s_i^-+c_{i-1}^-)},\quad d_i^-=\overline{(s_i^++c_{i-1}^+)}\cdot(s_i^-+c_{i-1}^-),$$

<33> $d_i^+ = \overline{(l_i \oplus h_{i-1})} \cdot (\overline{l_{i-1}} \cdot k_{i-1} + l_{i-1} \cdot \overline{h_{i-2}})$

<34> $d_i^- = (l_i \oplus h_{i-1}) \cdot \overline{(\overline{l_{i-1}} \cdot k_{i-1} + l_{i-1} \cdot \overline{h_{i-2}})}$

<35> $l_i = (a_i^+ + a_i^-) \oplus (b_i^+ + b_i^-)$, $h_i = a_i^- + b_i^-$, $k_i = a_i^+ + b_i^+$

<36> 만일 사용되는 수 체계의 진수(radix)를 증가시키면 부분곱들의 수는 더 감소하게 된다. 일반적으로 2^k진수를 사용하면 부분곱들의 수가 $\lceil \frac{m}{k} \rceil + 1$ 로 감소한다. 이용한 2^k진수 수 체계를 사용하는 m비트의 X*Y 곱셈의 경우 Y는 일반적으로 다음 수학식 6과 같이 표현된다.

<37> 【수학식 6】

$$Y = \sum_{i=0}^{\frac{m}{k}-1} D_i \cdot 2^{ki} = D_{\frac{m}{k}-1} D_{\frac{m}{k}-2} \cdots D_i \cdots D_1 D_0 = D$$

<38> 여기서,

<39> $$D_i = y_{ki-1} + (\sum_{j=1}^{k-1} y_{ki+j-1} \cdot 2^{j-1}) - y_{k(i+1)-1} \cdot 2^{k-1}$$

<40> 이다. 4진수 MBA의 경우 X에 곱해지는 Di는 {-2,-1,0,1,2}의 값들중 하나의 값을 갖기 때문에, 부분곱은 {-2X, -X, 0, X, 2X}의 값들중 하나가 된다. 이들 X의 배수들은 X를 왼쪽으로 자리를 이동시키는 연산만으로 구할 수 있기 때문에, 하드웨어 측면에서 거의 부담이 없다. 그러나 사용되는 수 체계의 진수가 높아질수록 하드웨어 부담은 점점 더 늘어난다. 예컨데, 일반적으로 2^k진수를 갖는 곱셈의 경우 D_i는 {-2^{k-1}, $-2^{k-1}+1$, · ··, 0, · ··, $2^{k-1}-1$, 2^{k-1}}의 값들중 하나의 값을 갖는다. 즉 Y의 각 묶음과의 곱에서 나올 수 있는 X의 배수는 2^k+1개로서, {$-2^{k-1}X$, $(-2^{k-1}+1)X$, · ··, 0X, · ··, $(2^{k-1}-1)X$, $2^{k-1}X$}

가 있게 된다. 예를 들어, 64 진수의 경우 k는 6이고 총 65개의 X배수가 필요하게 된다. 이들 배수 중에서 2^iX와 같은 X의 배수는 단순히 X를 i비트만큼 왼쪽으로 이동하는 연산만으로 구할 수가 있다. 그러나 $(2^i+1)X$와 같은 홀배수의 X배수는 전술한 자리 이동과 같은 간단한 방법으로 구할 수 없다. 즉, 3X의 경우는 X+2X 와 같은 덧셈 연산이 별도로 요구된다.

<41> 결국, 잉여 이진 연산을 이용한 곱셈기에서 종래의 연산 방법을 이용할 경우, 전술한 별도의 연산 과정이 홀배수의 X마다 이루어져야 하므로 곱셈에 사용되는 수 체계의 진수가 높아질 수록 추가로 필요한 덧셈 연산이 늘어나게 되어 하드웨어의 부담이 증가하는 문제점이 있다.

【발명이 이루고자 하는 기술적 과제】

<42> 본 발명이 이루고자 하는 기술적 과제는, 잉여 이진수 연산을 부분곱 생성에 적용하여 하드웨어의 증가를 감소시킬 수 있는 잉여 이진수 연산을 채택한 디지털 곱셈 장치를 제공하는 데 있다.

<43> 본 발명이 이루고자 하는 다른 기술적 과제는, 상기 잉여 이진수 연산을 채택한 디지털 곱셈 장치에서 수행되는 디지털 곱셈 방법을 제공하는 데 있다.

【발명의 구성 및 작용】

<44> 상기 과제를 이루기 위해, 2^k진수 수 체계를 사용하여 두 수(X 및 Y)를 곱셈하는 본 발명에 의한 디지털 곱셈 장치는, m비트의 상기 Y를 m/k 디지트의 $D(= D_{m/k-1}D_{m/k-2} \ldots D_i \ldots D_1D_0)$로 데이타 변환하는 데이타 변환부와, 상기 데이타 변환부에서 변환된 Y의 각 디지트 Di를 기본 배수의 계수들의 조합으로 변환하고, 변환된 조합을 상기 X와

승산하고, 승산된 결과를 잉여 이진수 형태의 부분곱으로서 출력하는 부분곱 산출부와, 변환된 상기 Y의 모든 디지트들에 대한 상기 부분곱들을 가산하는 잉여 이진수 가산부 및 잉여 이진수 형태의 상기 가산된 결과를 일반 이진수 형태로 변환하고, 변환된 일반 이진수 형태의 결과를 상기 두 수들의 곱셈 결과로서 출력하는 RB-NB 변환부로 구성되는 것이 바람직하다.

<45> 상기 다른 과제를 이루기 위해, 2^k진수 수 체계를 사용하여 두 수(X 및 Y)를 곱셈하는 본 발명에 의한 디지털 곱셈 방법은, m비트의 상기 Y를 m/k 디지트의 D(= $D_{m/k-1}D_{m/k-2}$ .... D_i ... D_1D_0)로 데이타 변환하는 단계와, 상기 Y의 각 디지트 Di를 기본 배수의 계수들의 조합으로 변환하고, 변환된 조합을 상기 X와 승산하여 잉여 이진수 형태의 부분곱을 구하는 단계와, 변환된 상기 Y의 모든 디지트들에 대한 상기 부분곱들을 가산하는 단계 및 잉여 이진수 형태의 상기 가산된 결과를 일반 이진수 형태로 변환하여 상기 두 수들의 곱셈 결과를 구하는 단계로 이루어지는 것이 바람직하다.

<46> 이하, 본 발명에 의한 잉여 이진수 연산을 채택한 디지털 곱셈 장치의 구성 및 동작을 첨부한 도면들을 참조하여 다음과 같이 설명한다.

<47> 도 1은 본 발명에 의한 잉여 이진수 연산을 채택한 디지털 곱셈 장치의 개략적인 블럭도로서, 데이타 변환부(10), 부분곱 산출부(12), 잉여 이진수 가산부(14) 및 RB(Redundant Binary)-NB(Normal Binary) 변환부(16)로 구성된다.

<48> 도 1에 도시된 본 발명에 의한 디지털 곱셈 장치는, 2^k진수 수 체계를 사용하여 두 수들 X 및 Y를 곱한다. 이를 위해, 먼저 데이타 변환부(10)는 m비트의 Y를 예를 들면 수학식 6과 같이 m/k 디지트의 D로 데이타 변환하고, 변환된 m/k 디지트의 데이타 D를

부분곱 산출부(12)로 출력한다.

<49> 부분곱 산출부(12)는 데이타 변환부(10)로부터 출력되는 D의 각 디지트 Di를 기본 배수들의 계수들의 조합으로 변환하고, 변환된 조합을 소정 비트수를 갖는 X와 승산하고, 승산된 결과를 수학식 3 및 4들의 설명에서 전술한 잉여 이진수 형태의 부분곱으로서 잉여 이진수 가산부(14)로 출력한다. 결국, 부분곱 산출부(12)로부터 $\lceil \frac{m}{k} \rceil + 1$ 개의 부분곱들이 잉여 이진수 가산부(14)로 출력된다.

<50> 잉여 이진수 가산부(14)는 데이타 변환부(10)에서 변환된 Y 즉, D의 모든 디지트들에 대한 $\lceil \frac{m}{k} \rceil + 1$ 개의 잉여 이진수 형태의 부분곱들을 가산하고, 가산된 결과를 RB-NB 변환부(16)로 출력한다. 이 때, RB-NB 변환부(16)는 잉여 이진수(RB:Redundant Binary) 형태의 가산된 결과를 입력하여 일반 이진수(NB:Normal Binary) 형태로 변환하고, 그 변환된 일반 이진수 형태의 가산된 결과를 두 수들 X와 Y의 곱셈 결과로써 출력단자 OUT를 통해 출력한다.

<51> 도 2는 도 1에 도시된 부분곱 산출부(12)의 본 발명에 의한 바람직한 실시예의 블럭도로서, 기본 배수 결정부(20), 제1 ~ 제m/k 멀티플렉서들(30, 32, ... 및 34), 제1 ~ 제m/k 논리 조합부들(40, 42, ... 및 44) 및 제어부(50)로 구성된다.

<52> 도 2에 도시된 기본 배수 결정부(20)는 m비트의 Y를 재부호화하여 비트 분할하고, 분할된 비트 묶음들을 소정 가중치와 각각 승산하고, 승산된 결과를 가산하여 기본 배수의 계수를 결정하고, 이와 같이 결정된 기본 배수의 계수들을 X와 승산하여 제1 ~ 제m/k 멀티플렉서들(30, 32, ... 및 36)로 출력한다. 각 멀티플렉서(30, 32, ... 또는 34)는 기본 배수 결정부(20)로부터 출력되는 e개의 기본 배수들중에서 두 개를 선택 신호(S1, S2, ... 또는 Sm/k)에 응답하여 선택하고, 선택된 결과를 해당하는 논리 조합부

(40, 42, ... 또는 44)로 출력한다. 여기서, 각 멀티플렉서(30, 32, ... 또는 34)로부터 출력되는 두개의 선택 결과들은 각각 기본 배수이다. 제1 ~ 제m/k 논리 조합부들(40, 42, ... 및 44) 각각은 대응하는 멀티플렉서(30, 32, ... 또는 34)에서 선택된 2개의 기본 배수들을 논리 조합하여 잉여 이진수 형태의 부분곱으로서 출력단자 OUT1, OUT2, ... 또는 OUTm/k를 통해 잉여 이진수 가산부(14)로 출력한다. 여기서, 제어부(50)는 데이타 변환부(10)에서 변환된 Y 즉, D의 모든 디지트들 m/k개를 입력단자 IN을 통해 입력하여 D의 각 디지트에 대한 기본 배수들의 조합을 생성하고, 생성된 조합에 상응하여 멀티플렉서에서 해당하는 기본 배수들이 선택될 수 있도록 선택 신호들(S1, S2, ... 및 Sm/k)을 발생한다.

<53> 본 발명의 이해를 돕기 위해 k=6라고 가정하여 도 1 및 도 2에 도시된 회로들의 동작과 그의 바람직한 실시예들에 대해서 다음과 같이 살펴본다.

<54> 64진수의 수 체계를 사용하여 두 수들 X와 Y를 곱셈할 때, 곱셈에 필요한 X배수는 총 65개로서 {-32X, -31X, ... , 0, ..., 31X, 32X} 이다. 이 때, 본 발명에 의한 다음 표 2를 이용해서 이들 65개의 배수들을 {0, X, 2X, 3X, 4X, 8X, 16X, 24X 및 32X}의 9(e=9)개 기본 배수들의 조합으로 표현할 수 있다. 여기서, 기본 배수들을 생성하는 세부적인 방법은 후술된다.

<55> 표 2에서, 배수의 계수(n)는 Di를 의미하고, A와 B에서 편의상 X는 모두 생략되었고 밑줄은 해당 배수의 각 비트를 모두 반전시킨 것이고 0's 나 1's 는 모든 비트가 0이거나 1인 경우를 나타낸다. 단, 여분 비트(extra bit)의 숫자들은 한 비트로서 2의 보수 연산 때문에 생긴 보정 비트(EX^+,EX^-)이다.

<56> 일반 이진수(normal binary number) 형태의 X배수들은 표 2에서와 같은 방법으로

몇 개의 일반 이진수 기본 배수(fundamental multiple)들의 조합만으로 된 잉여 이진수로 변환될 수 있다. 예를 들어, -20X를 잉여 이진수로 표현하는 과정은 다음 수학식 7과 같다.

<57> 【수학식 7】

$$-20X=-16X-4X=(\overline{16X}+1)-4X=(\overline{16X}-4X)+1\equiv(\overline{16X},4X)+(1,0)$$

<58> 【표 2】

n in nX	(A,B)	extra bit	n in nX	(A,B)	extra bit	n in nX	(A,B)	extra bit	n in nX	(A,B)	extra bit	extra bit
0	0's, 1's	0, 0	8	0's, 8	0, 1	16	0's, 16	0, 1	24	0's, 24	0, 1	
1	1, 1's	0, 0	9	1, 8	0, 1	17	1, 16	0, 1	25	1, 24	0, 1	
2	2, 1's	0, 0	10	2, 8	0, 1	18	2, 16	0, 1	26	2, 24	0, 1	
3	3, 1's	0, 0	11	3, 8	0, 1	19	3, 16	0, 1	27	3, 24	0, 1	
4	4, 1's	0, 0	12	4, 8	0, 1	20	4, 16	0, 1	28	4, 24	0, 1	
4*	4, 8	0, 0	12*	4, 16	0, 0	20*	4, 24	0, 0	28*	4, 32	0, 0	
5	3, 8	0, 0	13	3, 16	0, 0	21	3, 24	0, 0	29	3, 32	0, 0	
6	2, 8	0, 0	14	2, 16	0, 0	22	2, 24	0, 0	30	2, 32	0, 0	
7	1, 8	0, 0	15	1, 16	0, 0	23	1, 24	0, 0	31	1, 32	0, 0	
8	0's, 8	0, 0	16	0's, 16	0, 0	24	0's, 24	0, 0	32	0's, 32	0, 0	
n in nX	(A,B)	extra bit	n in nX	(A,B)	extra bit	n in nX	(A,B)	extra bit	n in nX	(A,B)	extra bit	extra bit
0	1's, 0's	0, 0	-8	8, 0's	1, 0	-16	16, 0's	1, 0	-24	24, 0's	1, 0	
-1	1's, 1	0, 0	-9	8, 1	1, 0	-17	16, 1	1, 0	-25	24, 1	1, 0	
-2	1's, 2	0, 0	-10	8, 2	1, 0	-18	16, 2	1, 0	-26	24, 2	1, 0	
-3	1's, 3	0, 0	-11	8, 3	1, 0	-19	16, 3	1, 0	-27	24, 3	1, 0	
-4	1's, 4	0, 0	-12	8, 4	1, 0	-20	16, 4	1, 0	-28	24, 4	1, 0	
-4*	8, 4	0, 0	-12*	16, 4	0, 0	-20*	24, 4	0, 0	-28*	32, 4	0, 0	
-5	8, 3	0, 0	-13	16, 3	0, 0	-21	24, 3	0, 0	-29	32, 3	0, 0	
-6	8, 2	0, 0	-14	16, 2	0, 0	-22	24, 2	0, 0	-30	32, 2	0, 0	
-7	8, 1	0, 0	-15	16, 1	0, 0	-23	24, 1	0, 0	-31	32, 1	0, 0	
-8*	8, 0's	0, 0	-16*	16, 0's	0, 0	-24*	24, 0's	0, 0	-32	32, 0's	0, 0	

<59> 예컨데, 수학식 7로부터 알 수 있듯이, $(\overline{16X},4X)$가 일반 이진수 형태의 부분곱인 -20X의 잉여 이진수 형태의 표현이며, (1,0)은 extra bit로서 2의 보수 연산 때문에 생긴 보정

비트이다. 64진수의 예에서는, 필요한 9개의 X 기본 배수들중 여분의 덧셈 연산을 필요로 하는 것은 3X뿐이다. 즉, 24X의 경우는 3X를 왼쪽으로 3비트 이동시킴으로써 구할 수 있고, 나머지 X기본 배수들은 X를 왼쪽으로 적절한 비트 수 만큼 이동시키면 구할 수 있으나 3X는 2X와 X를 더하는 여분의 덧셈 연산을 필요로 한다.

<60> 표 2에서 nX의 n이 양수인 경우와 음수인 경우를 비교하면 잉여 이진수 표현의 두 개의 자리 A와 B를 단순히 치환한 것임을 알 수 있다. 또한, n이 양수인 경우 (A,B)쌍을 보면 B는 항상 모든 비트를 반전함을 알 수 있고 A는 앞선 다섯줄들은 반전을 하지 않고 아래 다섯줄들은 모든 비트를 반전함을 알 수 있다. n이 음수인 경우는 A 및 B를 서로 바꾸면 전술한 동일한 규칙이 성립함을 알 수 있다. 그리고, n이 양수인 경우 표 2에서 B는 다섯줄씩 같은 기본 배수를 사용하게 됨을 알 수 있고 A는 10개의 줄마다 사용되는 기본 배수의 패턴을 반복함을 알 수 있다. 그리고 n이 음수이면 A와 B가 바뀌고 B는 항상 각 비트를 반전한다는 사실들을 이용하면, A 및 B는 다음 표 3과 같이 정리될 수 있다. 표 3은 잉여 이진수 형태의 부분곱(partial product)을 만들기 위한 진리표이다.

<61> 표 3에서 A_{Di}, B_{Di}, A_i, B_i 는 A_D, B_D, A, B의 비트들 각각을 나타내고, N은 X 의 배수가 음(1)인지 양(0)인지를 나타내는 변수이고, I_A는 A로 할당될 배수를 반전시킬지(1) 아닐지(0)를 나타내는 변수에 해당하고, A_D 및 B_D들은 후술되는 중간 변수들에 각각 해당한다. 예를 들어 표 2에서 n=11의 경우 n이 양이므로 N은 0이 되고 A가 3X이므로 I_A는 0이 되지만, -22X 의 경우 n이 음이므로 N은 1이고 A가 24의 반전 형태이므로 I_A는 1이 된다. A_D 및 B_D는 일반 이진수 배수를 잉여 이진수 배수로 표현하기 위해서 9개의 기본 배수들중에서 선택된 기본 배수에 해당하며, 이러한 중간 변수를 처리하여 최종 A 및 B 값을 구한다. 예를 들면, 표 2에서 13X의 경우

A_D 및 B_D는 각각 3X와 16X에 해당하고 N과 I_A는 각각 0, 1이 되서 A, B는 최종적으로 $\overline{3X}$, $\overline{16X}$가 된다. 또 -10X의 경우 A_D 및 B_D는 각각 2X 및 8X 가 되지만 N 과 I_A는 각각 1 및 0이 되어서 A 및 B는 최종적으로 8X의 반전된 값 및 2X가 된다.

<62> 【표 3】

0	0	0	0	0	1
0	0	0	1	0	0
0	0	1	0	1	1
0	0	1	1	1	0
0	1	0	0	1	1
0	1	0	1	1	0
0	1	1	0	0	1
0	1	1	1	0	0
1	0	0	0	1	0
1	0	0	1	1	1
1	0	1	0	0	0
1	0	1	1	0	1
1	1	0	0	0	0
1	1	0	1	0	1
1	1	1	0	1	0
1	1	1	1	1	1

<63> 표 2에서, n이 4, 8, 12, 16, 20, 24, 28, -4, -8, -12, -16, -20, -24, -28 일 경우는 같은 일반 이진수 배수라도 다른 잉여 이진수 표현들을 갖는다. 한편, 표 3은 다음 수학식 8과 같은 논리식으로 표현할 수 있다.

<64> 【수학식 8】

$$A_i=(N\oplus I_A)\oplus A_{Di} \ , \ B_i=\overline{(N\oplus B_{Di})}$$

<65> 수학식 8을 수행하는 본 발명에 의한 바람직한 실시예의 구성 및 동작을 살펴보면 다음과 같다.

<66> 도 3은 도 2에 도시된 멀티플렉서들중 하나 및 그에 대응하는 논리 조합부의

본 발명에 의한 바람직한 일실시예의 회로도로서, 멀티플렉서(60) 및 논리 조합부(70)로 구성된다.

<67> 도 3에 도시된 멀티플렉서(60)는 제1 ~ 제10 NMOS 트랜지스터들(MN1, MN2, MN3, MN4, MN5, MN6, MN7, MN8, MN9 및 MN10)로 구성된다. 각 트랜지스터는 해당하는 선택신호(S1X, S2X, S3X, S4X, ZERO, S8X, S16X, S24X, S32X 또는 ONE)와 연결되는 게이트, 기본 배수의 계수들과 X의 승산된 결과들인 부분곱들중 해당하는 부분곱인 기본 배수(X, 2X, 3X, 4X, VSS, VDD, 32X, 24X, 16X 또는 8X)와 논리 조합부(70) 사이에 연결되는 드레인 및 소스를 갖는다. 논리 조합부(70)는 멀티플렉서(60)에서 선택된 결과들중 하나를 반전하는 제1 인버터(72), 선택된 결과들중 다른 하나를 반전하는 제2 인버터(76), 제1 인버터(72)의 출력($\overline{A_{Di}}$)과 Y(··y_{6j+5} y_{6j+4} y_{6j+3} y_{6j+2} y_{6j+1} y_{6j} y_{6j-1}··)의 최하위 유효 비트(x_{6j-1})로부터 3번째 비트(x_{6j+2})를 배타적 논리합하는 제1 배타적 논리합부(74) 및 제2 인버터(76)의 출력($\overline{B_{Di}}$)과 Y의 최상위 유효 비트(y_{6j+5})를 배타적 논리합하는 제2 배타적 논리합부(78)로 구성된다. 도 3에서, d_i^+ 및 d_i^-는 표 2에서 A 및 B의 각 비트들 A_i 및 B_i에 해당하고, y_{6j+2}은 수학식 8에서 $N \oplus I_A$에 해당하고, y_{6j+5}는 수학식 8의 N에 해당한다.

<68> 전술한 구성을 부연하면, 도 3에 도시된 멀티플렉서(60)는 좌, 우에 5개씩 n형 모스로 구현되는 5-to-1 먹스(multiplexor)를 두 개 마련하고 있다. 멀티플렉서(60)는 SnX를 선택 신호로 받아들여 nX입력을 선택하며, 기본 배수의 계수(n)은 1, 2, 3, 4, 8, 16, 24, 32값이고, 왼쪽 먹스의 입력은 X, 2X, 3X, 4X 및 VSS의 각 비트이고 오른쪽 먹스의 입력은 8X, 16X, 24X, 32X 및 VDD의 각 비트이다. VSS 와 VDD는 각각 논리값 0과 1에 해당하고 이들 입력 값을 선택하는 먹스의 선택 신호는 각각 ZERO와 ONE이다. A_{Di} 및

B_{Di}는 해당하는 5-to-1 먹스의 출력이고, d_i^+ 및 d_i^-는 일반 이진수 배수를 잉여 이진수 배수 (A, B)로 표현한 수의 i번째 짝으로서 수학식 8의 A_i 및 B_i에 각각 해당한다.

<69> 도 3에서 d_i^+ 및 d_i^-를 잉여 이진 가산부(14)로 그대로 출력하지 않고 다음 수학식 9와 같이 변형하여 출력할 수도 있다.

<70> 【수학식 9】

$$d_i^+\overline{d_i^-}\ ,\ \overline{(d_i^+ \oplus d_i^-)}$$

<71> 이와 같이, 잉여 이진 배수를 수학식 9와 같이 변형시켜 출력하는 이유는 후술되는 바와 같이 잉여 이진 형태의 부분곱들을 서로 덧셈하는 하드웨어를 간단하게 구현하기 위해서이다. 이를 위해, 도 3에 도시된 바와 같이, 논리 조합부(70)는 제1 및 제2 배타적 논리합부들(74 및 78)의 출력을 반전 논리합하는 제1 반전 논리합부(80), 제1 및 제2 배타적 논리합부들(74 및 78)의 출력을 논리곱하는 제1 논리곱부(82) 및 제1 논리곱부(82)의 출력과 제1 반전 논리합부(80)의 출력을 반전 논리합하는 제2 반전 논리합부(84)를 더 마련할 수도 있다.

<72> 한편, 잉여 이진 가산부(14)에서 2개의 잉여 이진 형태의 부분곱들을 가산하는 동작을 살펴보면 다음과 같다.

<73> 도 3에 도시된 회로를 통해서 구한 잉여 이진수 부분곱들인 (a_i^+ 및 a_i^-) 및 (b_i^+ 및 b_i^-)는 본 발명에 의한 다음 표 4의 규칙을 이용해서 더한다.

<74>

<74>1020000070631 2000/12/2

【표 4】

case	(a_i^+, a_i^-) (b_i^+, b_i^-)	h_{i-1}	(c_i^+, c_i^-)	(s_i^+, s_i^-)
1	(0,0) (1,1) (0,1) (1,1) (0,0) (1,0) (0,0) (0,0) (1,0) (1,1) (1,1) (0,1)	any	(0,0)	(0,0)
2	(0,1) (0,0) (0,1) (1,1)	1	(0,0)	(0,1)
	(0,0) (0,1) (1,1) (0,1)	0	(0,1)	(1,0)
3	(1,0) (1,0) (0,0) (1,1)	1	(1,0)	(0,1)
	(0,0) (1,1) (1,0) (1,0)	0	(0,0)	(1,0)
4	(0,1) (0,1)	any	(0,1)	(0,0)
5	(1,0) (1,0)	any	(1,0)	(0,0)

<75> 표 1에 도시된 종래의 방식과 표 4에 도시된 본 발명에 의한 방식상의 차이점은, 표 4에서는 잉여 이진수 입력이 (1,1)인 경우도 고려했을 뿐만 아니라 h_i가 1인 경우에 입력되는 부분곱들 (a_i^+ 및 a_i^-)와 (b_i^+ 및 b_i^-)의 값들도 다르게 정의되었다. 즉, 종래의 표 1에서 h_i는 -1이 캐리로서 전파될 가능성이 있는 입력값에서 1이 되었는데 본 발명에 의한 표 4에서는 1이 캐리로서 전파될 가능성이 있는 입력값에서 h_i가 1이 된다. h_i의 역할이나 중간 합(s_i^+ 및 s_i^-), 중간 캐리(c_i^+ 및 c_i^-)를 구하는 방식은 표 1의 종래의 방식과 동일하므로 표 4에 대한 상세한 설명을 생략한다. 표 4에 의한 두개의 잉여 이진수 부분곱들 (a_i^+ 및 a_i^-) 와 (b_i^+ 및 b_i^-)의 합 (d_i^+ 및 d_i^-)는 다음 수학식 10과 같이 구할 수 있다.

<76> 【수학식 10】

$$d_i^+ = s_i^+ \cdot \overline{s_i^-} \cdot \overline{c_{i-1}^+} \cdot \overline{c_{i-1}^-} + \overline{s_i^+} \cdot \overline{s_i^-} \cdot c_{i-1}^+ \cdot \overline{c_{i-1}^-}, \quad d_i^- = \overline{s_i^+} \cdot s_i^- \cdot \overline{c_{i-1}^+} \cdot \overline{c_{i-1}^-} + \overline{s_i^+} \cdot \overline{s_i^-} \cdot \overline{c_{i-1}^+} \cdot$$

<77> $$d_i^+ = (l_i \oplus h_{i-1}) \cdot (\overline{l_{i-1}} \cdot k_{i-1} + l_{i-1} \cdot h_{i-2}) \equiv \alpha_i \cdot \overline{\beta_{i-1}}$$

<78> $$d_i^- = \overline{(l_i \oplus h_{i-1})} \cdot (\overline{l_{i-1}} \cdot \overline{k_{i-1}} + l_{i-1} \cdot \overline{h_{i-2}}) \equiv \overline{\alpha_i} \cdot \beta_{i-1}$$

<79> $$l_i = (a_i^+ \oplus a_i^-) \oplus (b_i^+ \oplus b_i^-)$$

<80> $$h_i = a_i^+ \cdot \overline{a_i^-} + b_i^+ \cdot \overline{b_i^-}$$

<81> $$k_i = \overline{(a_i^+ \oplus a_i^-)} + (a_i^+ \cdot \overline{a_i^-}) \cdot (b_i^+ \cdot \overline{b_i^-})$$

<82> 수학식 10을 회로로 구현하기 위해서는 다음과 같은 규칙성을 이용한다. 먼저, 입력값인 (a_i^+ 및 a_i^-) 및 (b_i^+ 및 b_i^-)를 이용해서 중간 변수인 l_i, h_i, k_i값들을 구하는 식은 $x_i^+ \oplus x_i^-$와 $x_i^+ \cdot x_i^-$(여기서, x는 a 또는 b)의 조합으로 이루어져 있다. 그러므로, 전술한 바와 같이 도 3에서 잉여 이진수의 부분곱을 만들어 낼 때 d_i^+ 및 d_i^-를 그대로 잉여 이진수 가산부(14)로 출력하지 않고, 추가로 논리 게이트들(80, 82 및 84)들을 더 마련하여 만든 수학식 9에 값들을 잉여 이진수 가산부(14)로 출력하였다. 이러한 규칙성을 이용하는 잉여 이진수 가산부(14)는 제1 ~ 제 $\lceil \log_2(\lceil \frac{m}{k} \rceil + 1) \rceil$ 잉여 이진수 가산단들(미도시)로 구성되고, 각 잉여 이진수 가산단(미도시)은 각각이 상보형 모스로 구현되는 잉여 이진수 덧셈기인 소정수개의 가산기들(미도시)로 구성된다. 이와 같이, 본 발명에 의한 표 4는 잉여 이진수 입력값이 (1,1)인 경우도 고려하였기 때문에, 종래의 표 1에서 (1,1)을 (0,0)으로 만들어주는 하드웨어의 부담을 줄일 수 있도록 한다.

<83> 이하, 소정수개의 가산기들(미도시) 각각의 본 발명에 의한 바람직한 일실시예의 구성 및 동작을 다음과 같이 첨부한 도면들을 참조하여 설명한다.

<84> 도 4는 가산될 부분곱들중 하나가 (0, 0)인 경우, 간략화된 가산기의 본 발명에 의

한 바람직한 일실시예의 회로도로서, 제2 및 제3 논리곱부들(90 및 96), 제3, 제4, 제5 및 제6 반전 논리합부들(92, 94, 98 및 100)로 구성된다.

<85> 도 4에 도시된 가산기의 제3 반전 논리합부(92)는 해당하는 논리 조합부에 제2 반전 논리합부의 출력($\overline{I_i}$)를 이전의 캐리 변수(h_{i-1})와 반전 논리합하여 수학식 10에 제시된 β_i로서 출력하고, 제2 논리곱부(90)는 제3 반전 논리합부(92)의 입력들을 논리곱하고, 제4 반전 논리합부(94)는 제3 반전 논리합부(92)의 출력과 제2 논리곱부(90)의 출력을 반전 논리합하여 수학식 10에 제시된 $\overline{\alpha_i}$로서 출력한다. 이 때, 제5 반전 논리합부(98)는 제3 반전 논리합부(92)에서 이전에 반전 논리합한 결과(β_{i-1})와 제4 반전 논리합부(94)의 출력을 반전 논리합하여 $d_i^+ \cdot \overline{d_i^-}$로서 출력하고, 제3 논리곱부(96)는 제3 반전 논리합부(92)에서 이전에 반전 논리합한 결과(β_{i-1})와 제4 반전 논리합부(94)의 출력을 논리곱하고, 제6 반전 논리합부(100)는 제5 반전 논리합부(98)의 출력과 제3 논리곱부(96)의 출력을 반전 논리합하여 $\overline{d_i^+ \oplus d_i^-}$로서 출력한다. 이와 같이, 잉여 이진수 형태의 부분곱들중 하나의 잉여 이진수 입력인 (a_i^+, a_i^-)가 (0,0)인 경우에는 도 4에 도시된 바와 같이 가산기가 간단히 구현될 수 있다.

<86> 이하, 어떠한 형태의 잉여 이진수들도 가산할 수 있는 본 발명에 의한 가산기의 바람직한 다른 실시예의 구성 및 동작을 살펴보면 다음과 같다.

<87> 도 5는 본 발명에 의한 가산기의 바람직한 다른 실시예의 회로도로서, 제4, 제5 및 제6 인버터들(116, 120 및 122), 상보형 모스 인버터(118)를 구성하는 p형 및 n형 모스 트랜지스터들, 제1, 제2, 제3, 제4 및 제5 전송 게이트들(108, 124, 126, 128 및 130), 제4 및 제5 논리곱부들(110 및 134), 제7, 제8, 제9 및 제10 반전 논리합부들(112, 114, 132 및 136)로 구성된다.

<88> 도 5를 참조하면, 제4 논리곱부(110)는 하나의 논리 조합부에 제1 반전 논리합부의 출력($a_i^+\overline{a_i^-}$)과 다른 하나의 논리 조합부에 제1 반전 논리합부의 출력($b_i^+\overline{b_i^-}$)을 논리곱하고, 제7 반전 논리합부(112)는 제4 논리곱부(110)의 입력들을 반전 논리합하여 $\overline{h_i}$ 로서 출력하며, 제8 반전 논리합부(114)는 제4 논리곱부(110)의 출력과 하나의 논리 조합부에 제2 반전 논리합부의 출력 $\overline{(a_i^+ \oplus a_i^-)}$ 을 반전 논리합하여 $\overline{k_i}$ 로서 출력한다. 이 때, 제3 인버터(116)는 하나의 논리 조합부에 제2 반전 논리합부의 출력 $\overline{(a_i^+ \oplus a_i^-)}$ 을 반전하며, 상보형 모스 인버터(118)는 하나의 논리 조합부에 제2 반전 논리합부의 출력 $\overline{(a_i^+ \oplus a_i^-)}$ 과 제3 인버터(116)의 출력 사이에 마련되어, 다른 하나의 논리 조합부에 제2 반전 논리합부의 출력 $\overline{(b_i^+ \oplus b_i^-)}$ 을 입력하여 반전하고, 반전된 결과를 l_i 로서 출력한다. 하나의 논리 조합부에 제2 반전 논리합부의 출력 $\overline{(a_i^+ \oplus a_i^-)}$ 및 제3 인버터(116)의 출력에 응답하여 제1 전송 게이트(108)는 상보형 모스 인버터(118)의 입력 $\overline{(b_i^+ \oplus b_i^-)}$ 을 인버터(120)의 입력(l_i)으로서 출력하고, 제4 인버터(120)는 상보형 모스 인버터(118)의 출력을 반전하여 출력하며, 제5 인버터(122)는 제7 반전 논리합부(112)의 이전 출력($\overline{h_{i-1}}$)을 반전하여 출력하고, 제2 전송 게이트(124)는 제5 인버터(122)의 출력을 제4 인버터(120)의 출력 및 상보형 모스 인버터(118)의 출력(l_i)에 응답하여 전송한다. 제3 전송 게이트(126)는 제7 반전 논리합부(112)의 이전 출력($\overline{h_{i-1}}$)을 상보형 모스 인버터(118)의 출력 및 제4 인버터(120)의 출력에 응답하여 전송하고, 제4 전송 게이트(128)는 제7 반전 논리합부(112)의 이전 출력($\overline{h_{i-1}}$)을 제4 인버터(120)의 출력 및 상보형 모스 인버터(118)의 출력(l_i)에 응답하여 전송하고, 제5 전송 게이트(130)는 제8 반전 논리합부(114)의 출력을 상보형 모스 인버터(118)의 출력(l_i) 및 제4 인버터(120)의 출력에 응답하여 전송한다. 한편, 제9 반전 논리합부(132)는 제2 및 제3 전송 게이트들(124 및 126)의 출력들($\overline{\alpha_i}$)

과 제4 및 제5 전송 게이트들(128 및 130)의 이전 출력들(β_{i-1})을 반전 논리합하고, 반전 논리합한 결과를 $d_i^+ \cdot \overline{d_i^-}$로서 출력한다. 이 때, 제5 논리곱부(134)는 제9 반전 논리합부(132)의 입력들을 논리곱하며, 제10 반전 논리합부(136)는 제5 논리곱부(134)의 출력과 제9 반전 논리합부(132)의 출력을 반전 논리합하고, 반전 논리합한 결과를 $\overline{d_i^+ \oplus d_i^-}$로서 출력한다.

<89> 하고 이하, 도 1에 도시된 잉여 이진수 연산 방식을 채택한 디지털 곱셈장치에서 수행되는 본 발명에 의한 디지털 곱셈 방법을 첨부한 도면들을 참조하여 다음과 같이 설명한다.

<90> 도 6은 본 발명에 의한 디지털 곱셈 방법을 설명하기 위한 플로우차트로서, 곱셈할 두 수들중 하나를 비트 변환하는 단계(제150 단계), 잉여 이진수 형태의 부분곱들을 구하는 단계(제152 ~ 제156 단계) 및 잉여 이진수 부분곱들을 가산하여 두 수(X, Y)의 곱셈 결과를 구하는 단계(제158 및 제160 단계들)로 이루어진다.

<91> 본 발명에 의한 디지털 곱셈 방법에서는 먼저 m비트의 Y를 수학식 6과 같이 m/k 디지트들로 이루어진 D로 데이타 변환한다(제150 단계). 제150 단계는 도 1에 도시된 데이타 변환부(10)에서 수행된다. 제150 단계후에, 도 2에 도시된 기본 배수 결정부(20)는 기본 배수의 계수들을 결정한다(제152 단계). 제152 단계를 세부적으로 살펴보면 다음과 같다.

<92> 도 7은 도 6에 도시된 기본 배수의 계수를 결정하는 단계(제152 단계)를 설명하기 위한 본 발명에 의한 플로우차트로서, 곱셈할 두 수들중 하나를 비트 분할하는 단계(제180 및 제182 단계) 및 분할된 비트를 가중치와 승산하여 기본 배수의 계수들을 구하는 단계(제184 및 제186 단계)로 이루어진다.

<93> 제150 단계후에, m비트인 Y를 한 비트씩 겹치면서 k+1비트씩 묶는다(제180 단계). 제180 단계후에, 최상위 유효 비트를 제외한 나머지 k비트들을 상위 t비트와 하위 s(≥ t)비트 즉, (t,s)로 분할한다(제182 단계). 여기서, k=s+t가 된다. 제182 단계후에, s개의 하위 비트들 각각을 소정 가중치들중 해당하는 가중치와 각각 승산하고, 승산된 결과들을 가산하여 s비트 묶음값을 구한다(제184 단계). 제184 단계후에, t개의 상위 비트들 각각을 가중치들중 해당하는 가중치와 각각 승산하고, 승산된 결과들을 가산하고, 가산된 결과를 2^s과 승산하여 t비트 묶음값을 구한다(제186 단계). 이 때, 기본 배수의 계수들은 s비트 묶음값들 및 t비트 묶음값들로부터 결정된다. 여기서, 제186 단계는 제184 단계보다 먼저 수행될 수도 있다.

<94> 이하, 2^k진수의 경우, 기본 배수의 계수를 결정하는 방법에 대한 본 발명에 의한 바람직한 일실시예를 첨부한 도면을 참조하여 다음과 같이 설명한다.

<95> 도 8은 도 6에 도시된 기본 배수의 계수들을 결정하기 위한 제152 단계의 본 발명에 의한 바람직한 일실시예의 플로우차트로서, Y_j(여기서, Y_j는 승수 Y의 j번째 k+1비트 묶음을 나타낸다.)의 하위 비트들을 이용하여 기본 배수의 계수들을 구하는 단계(제200 ~ 제206 단계) 및 Y_j의 상위 비트들을 이용하여 기본 배수의 계수들을 구하는 단계(제208 ~ 제214 단계)로 이루어진다.

<96> 먼저, Y_j의 하위 비트들의 수를 s라 하고 Y_j의 상위 비트들의 수를 t라 하고, k=t+s이고, $t=\lfloor \frac{k}{2} \rfloor$ (여기서, $\lfloor x \rfloor$ 는 x와 같거나 x보다 적은 가장 큰 정수를 의미한다.)라 하자. 즉, $t \leq s$ 가 된다.

<97> s비트인 이진수(s_grp_2)를 10진수로 표현한 값(s_grp_{10})을 초기화시킨다(제200 단

계). 제200 단계후에, s_grp_{10}에 대한 기본 배수의 계수값을 다음 수학식 11과 같이 구한다(제202 단계).

<98> 【수학식 11】

$$FMC[s\_grp_{10}]=s\_grp_2[0]+\sum_{j=1}^{t-1}s\_grp_2[j]*2^{j-1}$$

<99> 여기서, $s_grp_2[i]$는 s_grp_2의 i번째 비트를 나타내고, FMC는 기준 배수의 계수들의 배열을 나타낸다.

<100> 제202 단계후에, s_grp_{10}가 2^s보다 적은가를 판단한다(제204 단계). 만일, s_grp_{10}가 2^s보다 적으면 s_grp_{10}를 1만큼 증가시키고 제202 단계로 진행한다(제206 단계).

<101> 그러나, s_grp_{10}가 2^s보다 적지 않으면, t비트인 이진수(t_grp_2)를 10진수로 표현한 값(t_grp_{10})을 초기화시킨다(제208 단계). 제208 단계후에, t_grp_{10}에 대한 기본 배수의 계수값을 다음 수학식 12와 같이 구한다(제210 단계).

<102> 【수학식 12】

$$FMC[s\_grp_{10}+t\_grp_{10}]=2^s*(t\_grp_2[0]+\sum_{j=1}^{t-1}t\_grp_2[j]*2^j$$

<103> 여기서, $t_grp_2[i]$는 t_grp_2의 i번째 비트를 나타낸다.

<104> 제210 단계후에, t_grp_{10}가 2^t보다 적은가를 판단한다(제212 단계). 만일, t_grp_{10}가 2^t보다 적으면 t_grp_{10}를 1만큼 증가시키고 제210 단계로 진행한다(제214 단계). 그러나, t_grp_{10}가 2^t보다 적지 않으면 제152 단계를 종료한다.

<105> 수학식 11 및 12들에서 구한 FMC의 내용은 2^k에서 요구되는 기본 배수의 계수이지만, 같은 기본 배수의 계수가 여러번 중복해서 구해질 수 있다.

<106> 도 6에 도시된 제152 단계 및 도 7 및 도 8에 도시된 방법의 이해를 돕기 위해, k=6이라 가정하여 본 발명에 의한 기본 배수의 계수를 결정하는 방법을 다음과 같이 상세하게 설명한다.

<107> 먼저, Y를 64진수로 각 비트를 한 비트씩 겹치면서 7비트씩 묶어서 재 코딩(recoding)하면 다음 표 5와 같이된다. 즉, 표 5는 두 개의 이진수 곱셈 입력 값들 중 하나를 64진수로 재 부호화한 경우로서, 이진수의 비트 묶음을 64진수로 변환하는 표이다.

<108> 표 5에서 y6 비트는 n의 부호를 결정하고, 수학식 6으로부터 알 수 있듯이 하나의 n에 대해서 두 가지의 Y묶음이 나올 수 있는데 이들 중 몇몇은 회로 구현을 간단히 하기 위해서 서로 다르게 취급되어야 할 것을 표 5에서 박스 표시로서 구분하였다.

<109>

【표 5】

x6x5x4x3x2x1x0	n in nX	x6x5x4x3x2x1x0	n in nX	x6x5x4x3x2x1x0	n in nX	x6x5x4x3x2x1x0	n in nX
0000000	0	0011111	16	1111111	-0	1011110	-17
0000001	1	0100000		1111110		1011101	
0000010		0100001	17	1111101	-1	1011100	-18
0000011	2	0100010		1111100		1011011	
0000100		0100011	18	1111011	-2	1011010	-19
0000101	3	0100100		1111010		1011001	
0000110		0100101	19	1111001	-3	1011000	
0000111	4	0100110		1111000		1010111	-20
0001000		0100111	20	1110111	-4	1010110	
0001001	5	0101000		1110110		1010101	-21
0001010		0101001	21	1110101	-5	1010100	
0001011	6	0101010		1110100		1010011	-22
0001100		0101011	22	1110011	-6	1010010	
0001101	7	0101100		1110010		1010001	-23
0001110		0101101	23	1110001	-7	1010000	
0001111	8	0101110		1110000		1001111	-24
0010000		0101111	24	1101111	-8	1001110	
0010001	9	0110000		1101110		1001101	-25
0010010		0110001	25	1101101	-9	1001100	
0010011	10	0110010		1101100		1001011	-26
0010100		0110011	26	1101011	-10	1001010	
0010101	11	0110100		1101010		1001001	-27
0010110		0110101	27	1101001	-11	1001000	
0010111	12	0110110		1101000		1000111	-28
0011000		0110111	28	1100111	-12	1000110	
0011001	13	0111000		1100110		1000101	-29
0011010		0111001	29	1100101	-13	1000100	
0011011	14	0111010		1100100		1000011	-30
0011100		0111011	30	1100011	-14	1000010	
0011101	15	0111100		1100010		1000001	-31
0011110		0111101	31	1100001	-15	1000000	-32
		0111110		1100000			
		0111111	32	1011111	-16		

<110> Y_j에서 최상위 유효 비트(y6)을 제외한 나머지 비트들을 (t,s)=(3 비트, 3비트)로 나눈다. 여기서, 각 묶음값을 구하기 위해, 각 비트 별 가중치를 최소 유효자리부터 시작해서 예를 들면, 1, 1, 2, 4, 8, ..., 2^i 로 설정한다. 여기서, k=6이면, 가중치는 1, 1 및 2가 된다. 따라서, s비트 묶음의 비트가 '101'이라면 2·1 + 1·0 + 1·1이므로 s

비트 묶음값은 3이 된다. 그리고, t비트 묶음이 s비트 묶음보다 위의 자리수들이므로 적절한 가중치를 곱해야 하는데 2^s 을 t비트 묶음에서 구한 수에 곱한다. 즉, t비트 묶음이 '101'이라면, $(2\cdot 1 + 1\cdot 0 + 1\cdot 1)\times 2^3$이므로 t비트 묶음값은 24가 된다. 이와 같은 방법으로 구한 비트 묶음값들이 표 6에 나타나 있으며, 이러한 비트 묶음값들은 기본 배수 nX의 n에 해당한다.

<111> 표 6은 본 발명에 의해, 여러 개의 일반 이진수 배수들을 단지 몇 개만의 기본 배수들의 조합으로 표현하기 위해서 필요한 비트 묶음값들을 나타낸다.

<112> 【표 6】

y6	s 비트 묶음 (s=3)			s 비트 묶음 값	t 비트 묶음 (t=3)			t 비트 묶음 값
	y5	y4	y3		y2	y1	y0	
무시	0	0	0	0	0	0	0	0
	0	0	1	8	0	0	1	1
	0	1	0	8	0	1	0	1
	0	1	1	16	0	1	1	2
	1	0	0	16	1	0	0	2
	1	0	1	24	1	0	1	3
	1	1	0	24	1	1	0	3
	1	1	1	32	1	1	1	4

<113> 표 6으로부터 기본 배수는 0X, 1X, 2X, 3X, 4X, 8X, 16X, 24X, 32X의 9개가 됨을 알 수 있다.

<114> 한편, 제152 단계후에, 제어부(50)는 데이타 변환부(10)로부터 입력단자 IN을 통해 D를 입력하여, D의 각 디지트(Di)를 기본 배수의 계수들의 조합으로 변환한다(제154 단계). 제154 단계후에, 변환된 조합들 각각을 X와 승산하여 잉여 이진수 형태의 부분곱을 구한다(제156 단계). 이를 위해, 제어부(50)는 전술한 바와 같

이 기본 배수의 계수들의 조합에 따라 선택 신호들(S1, S2, ... 및 Sm/k)을 발생함으로서 기본 배수의 계수들의 조합과 X가 승산된 값들이 멀티플렉서들에서 선택되도록 한다. 제156 단계후에, 잉여 이진수 가산부(14)는 잉여 이진수 형태의 부분곱들을 가산한다(제158 단계). 제158 단계후에, 잉여 이진수 형태의 가산된 결과를 일반 이진수 형태로 변환하여 두 수들 X 및 Y의 최종 곱셈 결과를 구한다(제160 단계).

<115> 이하, k=6이라 하고, 승수 Y를 부호화해서 피승수 X와 곱한 일반 이진수로 된 부분곱을 기본 배수의 조합인 잉여 이진수로 된 부분곱으로 변환하는 본 발명에 의한 잉여 이진수 연산을 채택한 디지털 곱셈 방법의 바람직한 일실시예를 첨부한 도면을 참조하여 다음과 같이 설명한다.

<116> 도 9는 도 6에 도시된 제152 ~ 제156 단계들을 수행하는 본 발명에 의한 잉여 이진수 연산을 채택한 디지털 곱셈 방법의 바람직한 일실시예의 플로우차트로서, 기본 배수의 계수들을 결정하는 단계(제240 ~ 제250 단계), 잉여 이진수 형태의 부분곱(A,B)과 보정비트(EX^+,EX^-)를 구하는 단계(제252 ~ 제268 단계)로 이루어진다. 여기서, (A, B)는 Y_j를 부호화한 결과와 X를 곱한 일반 이진수로 된 부분곱의 잉여 이진수 표현이다.

<117> 도 9를 참조하면, 기본 배수의 계수들은 다음과 같이 결정된다(제240 ~ 제250 단계).

<118> 먼저, Y_j의 부호(SIGN) 비트인 최상위 유효 비트(MSB)가 '1' 인가를 판단한다(제240 단계). 만일, Y_j의 최상위 유효 비트(MSB)가 '1' 이면, Y_j의 각 비트를 반전시킨다(제242 단계). 그러나, Y_j의 최상위 유효 비트(MSB)가 '1'이 아니거나 제242 단계후에, Y_j의 최상위 유효 비트로부터 네번째 비트(I_A)가 '1' 인가를 판단한다(제244 단계). 만일, Y_j의 최

상위 유효 비트로부터 네번째 비트(I_A)가 '1' 이면, Y_j의 하위 s 비트들 각각을 반전시킨다(제246 단계). Y_j의 하위 s 비트들 각각을 해당하는 가중치들과 각각 승산하고, 승산된 결과를 가산하여 s비트 묶음값을 구하고, Y_j의 상위 t 비트들 각각을 해당하는 가중치들과 각각 승산하고, 승산된 결과를 2^3=8과 승산하여 t비트 묶음값을 구한다(제250 단계).

<119> 다음으로, 잉여 이진수 형태의 부분곱(A,B)과 보정 비트(EX^+,EX^-)를 구한다(제252 ~ 제268 단계).

<120> 제250 단계후에, I_A가 '1'인가를 판단한다(제252 단계). 만일, I_A가 '1' 이면, s비트 묶음값을 X와 승산하고, 승산된 결과를 반전시켜 A값을 구한다(제254 단계). 그러나, I_A가 '1' 이 아니면, s비트 묶음값을 X와 승산하여 A값을 구한다(제256 단계). 제254 단계 또는 제256 단계후에, t비트 묶음값을 X와 승산하고, 승산된 결과를 반전시켜 B값을 구한다(제258 단계). 제258 단계후에, I_A가 '1' 인가를 판단한다(제260 단계). 만일, I_A가 '1'이면, 보정 비트(EX^+,EX^-)를 (0,0)으로 설정한다(제262 단계). 그러나, I_A가 '1'이 아니면, 보정 비트(EX^+,EX^-)를 (0,1)로 설정한다(제264 단계). 제262 또는 제264 단계후에, 부호 비트인 Y_j의 최상위 유효 비트가 '1'인가를 판단한다(제266 단계). 만일, Y_j의 최상위 유효 비트가 '1'이면 (A,B)에서 A와 B를 바꾸고, (EX^+,EX^-)에서 EX^+와 EX^-를 바꾼다(제268 단계). 그러나, Y_j의 최상위 유효 비트가 '1'이 아니거나 제268 단계후에, 도 6에 도시된 제158 단계로 진행한다.

<121> 결국, 본 발명에서는 부분곱이 X의 홀배수들중 하나 예를 들면 27X인 경우 3X+24X와 같은 기본 배수들(3Y와 24Y)의 조합으로 구하기 때문에 두개의 일반 이진수들의 덧셈을 잉여 이진수로의 변환을 통해서 쉽게 이룰 수 있고 이 변환에는 하드웨어 부담이 거

의 없다. 즉 곱셈에 필요한 여러 개의 홀배수들을 그 홀배수들의 수보다 적은 개수의 기본 배수(fundamental multiple)들의 조합으로 표현할 수가 있다.

【발명의 효과】

<122>25(이상에서 설명한 바와 같이, 본 발명에 의한 잉여 이진수 연산을 채택한 디지털 곱셈 장치 및 방법은 부분곱을 만들어내는 부분에 잉여 이진수 개념을 도입하여 X의 배수 제를 일반 이진수가 아닌 잉여 이진수로 표현함으로써 하드웨어 부담을 크게 줄일 수가 있도록 하고, 부분곱을 만드는데 사용되는 진수(radix)를 널리 사용되는 MBA의 4에서 그 이상으로 확장시키면서도 하드웨어 부담을 최소화시킬 수 있으며, 진수가 올라가서 부분곱의 개수가 줄어들기 때문에 부분곱들을 가산하기 위한 하드웨어의 양도 줄일 수 있다. 게다가, 부분곱들을 더하는 잉여 이진수 덧셈기의 구조를 도 4 및 5들에 도시된 바와 같이 개선해서 곱셈 방식을 최적화시켰다. 그러므로, 곱셈기가 중요 구성 기능 블록인 여러 시스템들이 좀 더 간단하게 구현될 수 있도록 하는 효과가 있다.

【특허청구범위】

【청구항 1】

2^k진수 수 체계를 사용하여 두 수(X 및 Y)를 곱셈하는 디지털 곱셈 장치에 있어서,

m 비트의 상기 Y를 m/k 디지트의 D(= $D_{m/k-1}D_{m/k-2}$ .... D_i ... D_1D_0)로 데이타 변환하는 데이타 변환부;

상기 데이타 변환부에서 변환된 Y의 각 디지트 Di를 기본 배수의 계수들의 조합으로 변환하고, 변환된 조합을 상기 X와 승산하고, 승산된 결과를 잉여 이진수 형태의 부분곱으로서 출력하는 부분곱 산출부;

변환된 상기 Y의 모든 디지트들에 대한 상기 부분곱들을 가산하는 잉여 이진수 가산부; 및

잉여 이진수 형태의 상기 가산된 결과를 일반 이진수 형태로 변환하고, 변환된 일반 이진수 형태의 결과를 상기 두 수들의 곱셈 결과로서 출력하는 RB-NB 변환부를 구비하는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 장치.

【청구항 2】

제1 항에 있어서, 상기 부분곱 산출부는

상기 Y를 재부호화하여 상위 비트와 하위 비트로 분할하고, 분할된 상기 하위 비트들을 해당하는 소정 가중치들과 승산하고 승산된 결과들을 가산하여 상기 기본 배수의 계수로서 결정하고, 분할된 상기 상위 비트들을 상기 소정 가중치들과 승산하고 승산된 결과들을 가산후 2^k와 승산하여 상기 기본 배수 계수로서 결정하고, 결정된 상기 기본 배수의 계수들을 상기 X와 승산하여 상기 기본 배수로서 출력하는 기본 배수 결정부;

제1 ~ 제m/k 멀티플렉서들;

제 1 ~ 제m/k 논리 조합부들; 및

상기 데이타 변환부에서 변환된 Y의 모든 디지트들을 입력하여 각 디지트 D_i에 대한 상기 기본 배수의 계수들의 조합을 생성하고, 생성된 조합에 상응하여 선택 신호들을 발생하는 제어부를 구비하고,

상기 멀티플렉서들 각각은 상기 기본 배수들중 두 개를 상기 선택 신호에 응답하여 선택하고, 상기 논리 조합부는 대응하는 상기 멀티플렉서에서 선택된 결과들을 논리 조합하여 상기 잉여 이진수 형태의 부분곱으로서 상기 RB-NB 변환부로 출력하는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 장치.

【청구항 3】

제2 항에 있어서, 상기 각 논리 조합부는

상기 선택된 결과들중 하나를 반전하는 제1 인버터;

상기 선택된 결과들중 다른 하나를 반전하는 제2 인버터;

상기 제1 인버터의 출력과 상기 Y의 최하위 유효 비트로부터 소정수번째 비트를 배타적 논리합하는 제1 배타적 논리합부; 및

상기 제2 인버터의 출력과 상기 Y의 최상위 유효 비트를 배타적 논리합하는 제2 배타적 논리합부를 구비하고,

상기 제1 및 상기 제2 배타적 논리합부들에서 배타적 논리합한 결과들은 상기 잉여 이진수 형태의 부분곱에 해당하는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 장치.

【청구항 4】

제3 항에 있어서, 상기 각 논리 조합부는

상기 제1 및 상기 제2 배타적 논리합부들의 출력을 반전 논리합하는 제1 반전 논리합부;

상기 제1 및 상기 제2 배타적 논리합부들의 출력을 논리곱하는 제1 논리곱부; 및

상기 제1 논리곱부의 출력과 상기 제1 반전 논리합부의 출력을 반전 논리합하는 제2 반전 논리합부를 더 구비하는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 장치.

【청구항 5】

제3 항에 있어서, 상기 잉여 이진수 가산부는

제1 ~ 제 $\lceil \log_2(\lceil \frac{m}{k} \rceil + 1) \rceil$ 잉여 이진수 가산단들을 구비하고, 상기 각 잉여 이진수 가산단은 소정수개의 가산기들을 갖고, 상기 각 가산기는 잉여 이진수 형태의 두 개의 부분곱들 (a_i^+, a_i^-) 및 (b_i^+, b_i^-)을 아래와 같은 논리 조합식에 의해 가산하고, 가산된 결과 (d_i^+, d_i^-)를 출력하는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 장치.

$$d_i^+ = (l_i \oplus h_{i-1}) \cdot (\overline{l_{i-1}} \cdot k_{i-1} + l_{i-1} \cdot h_{i-2}) \equiv \alpha_i \cdot \overline{\beta_{i-1}}$$

$$d_i^- = \overline{(l_i \oplus h_{i-1})} \cdot (\overline{l_{i-1}} \cdot \overline{k_{i-1}} + l_{i-1} \cdot \overline{h_{i-2}}) \equiv \overline{\alpha_i} \cdot \beta_{i-1}$$

(여기서,

$$l_i=(a_i^+ \oplus a_i^-) \oplus (b_i^+ \oplus b_i^-),\quad h_i=a_i^+ \cdot \overline{a_i^-}+b_i^+ \cdot \overline{b_i^-},$$

$$k_i=\overline{(a_i^+ \oplus a_i^-)}+(a_i^+ \cdot \overline{a_i^-}) \cdot (b_i^+ \cdot \overline{b_i^-})$$

을 각각 의미한다.)

【청구항 6】

제5 항에 있어서, 상기 각 가산기는

해당하는 상기 논리 조합부에 상기 제2 반전 논리합부의 출력과 이전의 캐리 변수(h_{i-1})를 반전 논리합하는 제3 반전 논리합부;

해당하는 상기 논리 조합부의 상기 제2 반전 논리합부의 출력과 상기 이전의 캐리 변수를 논리곱하는 제2 논리곱부;

상기 제3 반전 논리합부의 출력과 상기 제2 논리곱부의 출력을 반전 논리합하는 제4 반전 논리합부;

상기 제3 반전 논리합부에서 이전에 반전 논리합한 결과와 상기 제4 반전 논리합부의 출력을 반전 논리합하는 제5 반전 논리합부;

상기 제3 반전 논리합부에서 이전에 반전 논리합한 결과와 상기 제4 반전 논리합부의 출력을 논리곱하는 제3 논리곱부;

상기 제5 반전 논리합부의 출력과 상기 제3 논리곱부의 출력을 반전 논리합하는 제6 반전 논리합부를 구비하고,

상기 제5 반전 논리합부의 출력은

$$d_i^+ \cdot \overline{d_i^-}$$

에 해당하고, 상기 제6 반전 논리합부의 출력은

$$\overline{d_i^+ \oplus d_i^-}$$

에 해당하는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 장치.

【청구항 7】

제5 항에 있어서, 상기 각 가산기는

하나의 상기 논리 조합부에 상기 제1 반전 논리합부의 출력과 다른 하나의 상기 논리 조합부에 상기 제1 반전 논리합부의 출력을 논리곱하는 제4 논리곱부;

상기 제4 논리곱부의 입력들을 반전 논리합하는 제7 반전 논리합부;

상기 제4 논리곱부의 출력과 상기 하나의 논리 조합부에 상기 제2 반전 논리합부의 출력을 반전 논리합하는 제8 반전 논리합부;

상기 하나의 논리 조합부에 상기 제2 반전 논리합부의 출력을 반전하는 제3 인버터;

상기 하나의 논리 조합부에 상기 제2 반전 논리합부의 출력과 상기 제3 인버터의 출력 사이에 마련되며, 상기 다른 하나의 논리 조합부에 상기 제2 반전 논리합부의 출력을 입력하여 반전하는 상보형 모스 인버터;

상기 상보형 모스 인버터의 출력을 반전하여 출력하는 제4 인버터;

상기 하나의 논리 조합부에 상기 제2 반전 논리합부의 출력 및 상기 제3 인버터의 출력에 응답하여 상기 상보형 모스 인버터의 입력을 상기 제4 인버터의 입력으로서 전

송하는 제1 전송 게이트;

상기 제7 반전 논리합부의 이전 출력을 반전하여 출력하는 제5 인버터;

상기 제5 인버터의 출력을 상기 제4 인버터의 출력 및 상기 상보형 모스 인버터의 출력에 응답하여 전송하는 제2 전송 게이트;

상기 제7 반전 논리합부의 이전 출력을 상기 상보형 모스 인버터의 출력 및 상기 출력을 제4 인버터의 출력에 응답하여 전송하는 제3 전송 게이트;

상기 제7 반전 논리합부의 이전 출력을 상기 제4 인버터의 출력 및 상기 상보형 모스 인버터의 출력에 응답하여 전송하는 제4 전송 게이트;

상기 제8 반전 논리합부의 출력을 상기 상보형 모스 인버터의 출력 및 상기 제4 인버터의 출력에 응답하여 전송하는 제5 전송 게이트;

상기 제2 및 상기 제3 전송 게이트들의 출력들과 상기 제4 및 상기 제5 전송 게이트들의 이전 출력들을 반전 논리합하는 제9 반전 논리합부;

상기 제9 반전 논리합부의 입력들을 논리곱하는 제5 논리곱부; 및

상기 제5 논리곱부의 출력과 상기 제9 반전 논리합부의 출력을 반전 논리합하는 제10 반전 논리합부를 구비하고,

상기 제9 반전 논리합부의 출력은

$$d_i^+ \cdot \overline{d_i^-}$$

에 해당하고, 상기 제10 반전 논리합부의 출력은

$$\overline{d_i^+ \oplus d_i^-}$$

에 해당하는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 장치.

【청구항 8】

2^k진수 수 체계를 사용하여 두 수(X 및 Y)를 곱셈하는 디지털 곱셈 방법에 있어서,

(a) m 비트의 상기 Y를 m/k 디지트의 $D(= D_{m/k-1}D_{m/k-2} \ldots D_i \ldots D_1D_0)$로 데이타 변환하는 단계;

(b) 상기 Y의 각 디지트 Di를 기본 배수의 계수들의 조합으로 변환하고, 변환된 조합을 상기 X와 승산하여 잉여 이진수 형태의 부분곱을 구하는 단계;

(c) 변환된 상기 Y의 모든 디지트들에 대한 상기 부분곱들을 가산하는 단계; 및

(d) 잉여 이진수 형태의 상기 가산된 결과를 일반 이진수 형태로 변환하여 상기 두 수들의 곱셈 결과를 구하는 단계를 구비하는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 방법.

【청구항 9】

제8 항에 있어서, 상기 (b) 단계는

(b1) 상기 기본 배수의 계수들을 결정하는 단계;

(b2) 상기 Di를 상기 기본 배수의 계수들의 조합으로 변환하는 단계; 및

(b3) 변환된 조합들 각각을 상기 X와 승산하여 상기 잉여 이진수 형태의 부분곱을 구하는 단계를 구비하는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 방법.

【청구항 10】

제9 항에 있어서, 상기 (b1) 단계는

상기 m비트인 Y를 한 비트씩 겹치면서 k+1비트씩 묶는 단계;

k+1비트 묶음에서 최상위 유효 비트를 제외한 나머지 k비트를 상위 t비트와 하위 s(≥t)비트로 분할하는 단계;

s 개의 하위 비트들 각각을 소정 가중치들중 해당하는 가중치와 각각 승산하고, 승산된 결과들을 가산하여 s비트 묶음값을 구하는 단계; 및

t개의 상위 비트들 각각을 상기 가중치들중 해당하는 상기 가중치와 각각 승산하고, 승산된 결과들을 가산하고, 가산된 결과를 2^s과 승산하여 t비트 묶음값을 구하는 단계를 구비하고,

상기 기본 배수의 계수들은 상기 s비트 묶음값들 및 상기 t비트 묶음값들로부터 결정되는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 방법.

【청구항 11】

제10 항에 있어서, 상기 k가 6인 경우, 상기 해당하는 가중치는 1, 1 및 2이고, 기본 배수 계수들은 0, 1, 2, 3, 4, 8, 16, 24 및 32에 해당하는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 방법.

【청구항 12】

제9 항에 있어서, 상기 (b1) 단계는

(b11) 상기 m비트인 Y를 한 비트씩 겹치면서 k+1비트씩 묶은 Y_j의 하위 s[여기서, s=k-t이고, $t=\lfloor \frac{k}{2} \rfloor$ ($\lfloor x \rfloor$ 는 x와 같거나 x보다 적은 가장 큰 정수를 의미한다.)

이다. s비트의 이진수(s_grp_2)를 10진수로 표현한 값(s_grp_{10})을 초기화시키는 단계;

(b12) 상기 기본 배수의 계수를 아래와 같이 구하는 단계;

$$FMC[s\_grp_{10}]=s\_grp_2[0]+\sum_{j=1}^{s-1}s\_grp_2[j]*2^{j-1}$$

(여기서, $s_grp_2[i]$는 s_grp_2의 i번째 비트를 나타낸다.)

(b13) s_grp_{10}가 2^s보다 적은가를 판단하는 단계;

(b14) s_grp_{10}가 2^s보다 적으면, s_grp_{10}를 1만큼 증가시키고 상기 (b12) 단계로 진행하는 단계;

(b15) s_grp_{10}가 2^s보다 적지 않으면, Y_j의 상위 t비트의 이진수(t_grp_2)를 10진수로 표현한 값(t_grp_{10})을 초기화시키는 단계;

(b16) 기본 배수의 계수값을 아래와 같이 구하는 단계;

$$FMC[s\_grp_{10}+t\_grp_{10}]=2^s*(t\_grp_2[0]+\sum_{j=1}^{t-1}t\_grp_2[j]*2^{j-1})$$

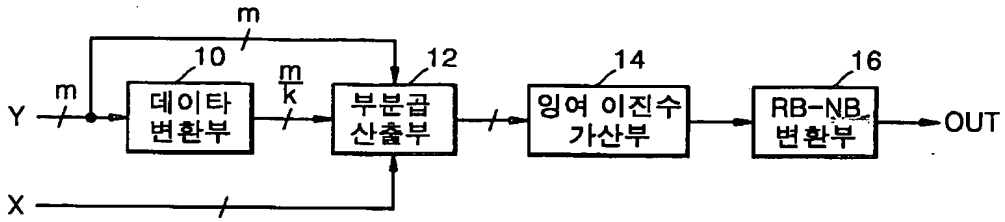
(여기서, $t_grp_2[i]$는 t_grp_2의 i번째 비트를 나타낸다.)

(b17) t_grp_{10}가 2^t보다 적은가를 판단하여, t_grp_{10}가 2^t보다 적지 않으면 상기 (b2) 단계로 진행하는 단계; 및

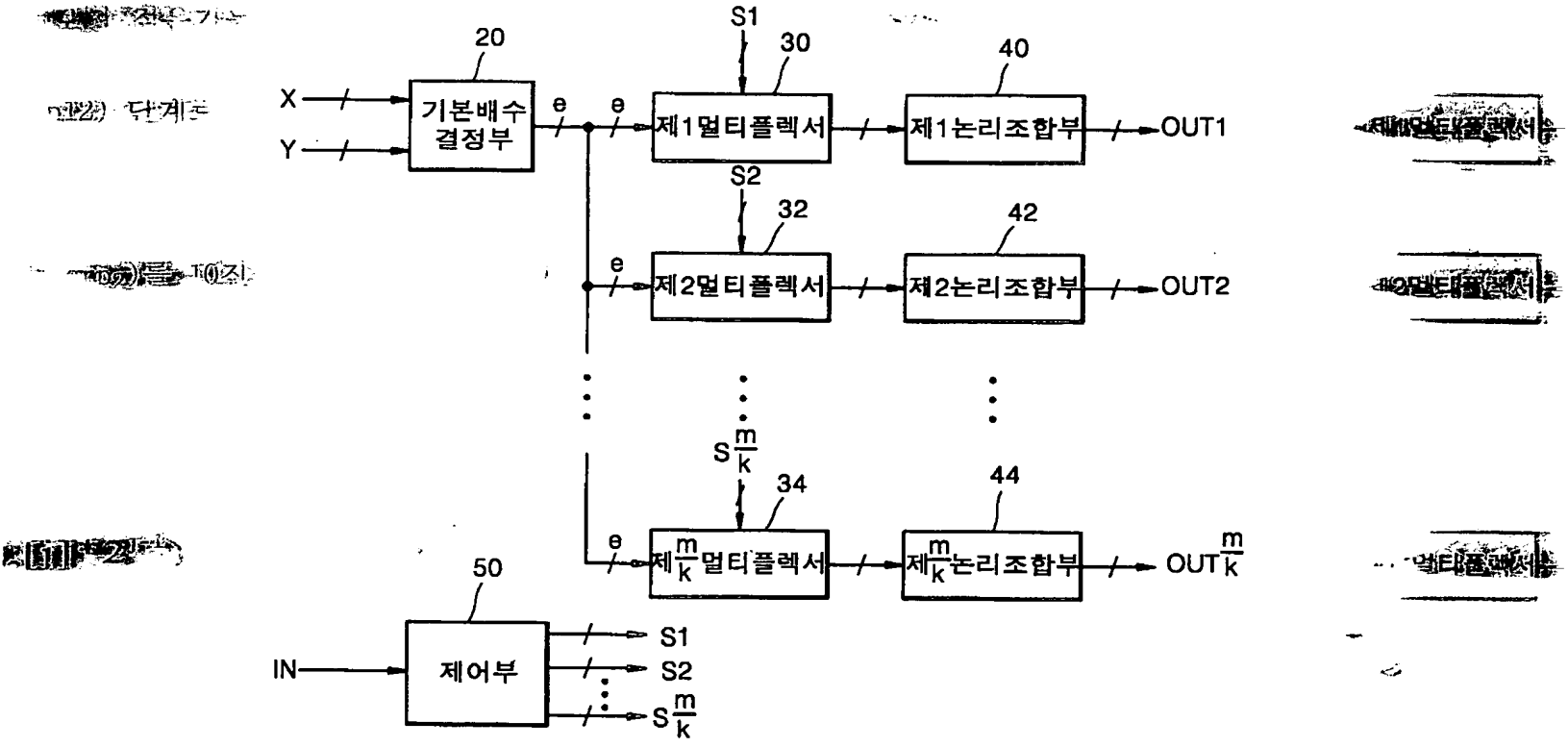
(b18) t_grp_{10}가 2^t보다 적으면 t_grp_{10}를 1만큼 증가시키고 상기 (b16) 단계로 진행하는 단계를 구비하는 것을 특징으로 하는 잉여 이진수 연산을 채택한 디지털 곱셈 방법.

【도면】

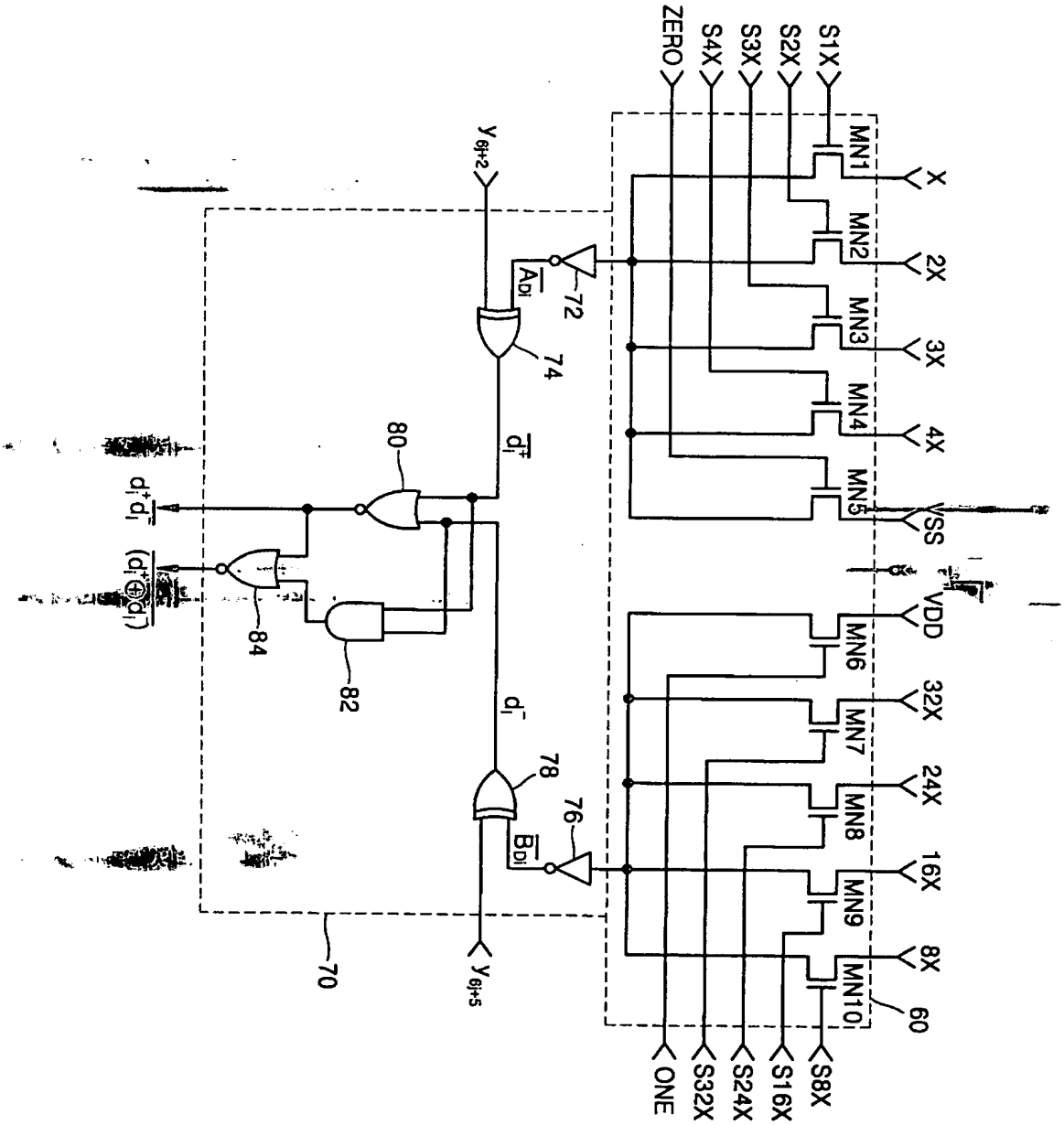
【도 1】


m
10
m
k
12
14
16
m
Y
데이타
변환부
부분곱
산출부
잉여 이진수
가산부
RB-NB
변환부
OUT
X

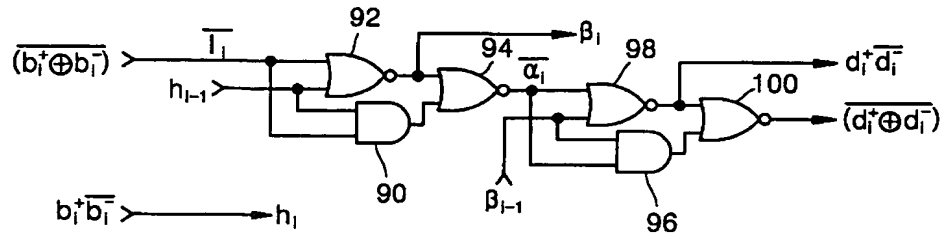
【도 2】


S1
20
30
40
X
Y
기본배수
결정부
e
e
제1멀티플렉서
제1논리조합부
OUT1
S2
32
42
e
제2멀티플렉서
제2논리조합부
OUT2
S m/k
34
44
e
제 m/k 멀티플렉서
제 m/k 논리조합부
OUT m/k
50
IN
제어부
S1
S2
S m/k

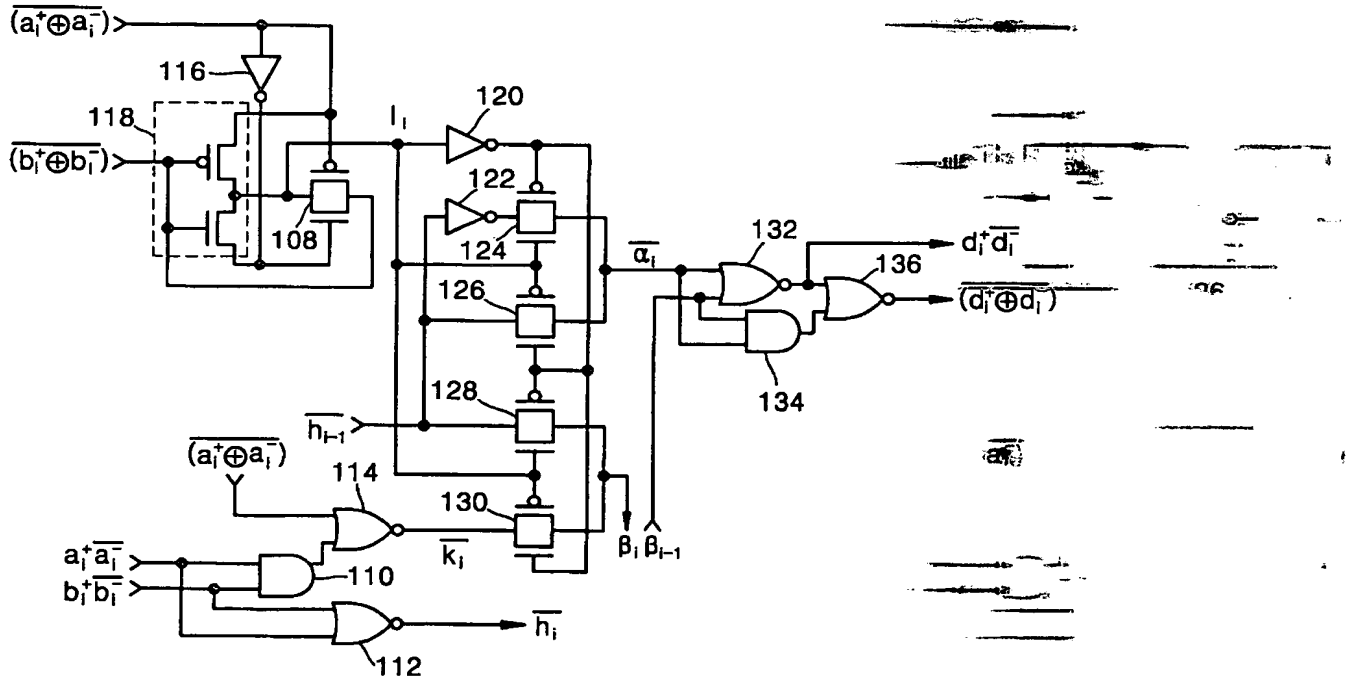
【도 3】


ZERO
S4X
S3X
S2X
S1X
MN1
MN2
MN3
MN4
MN5
MN6
MN7
MN8
MN9
MN10
X
2X
3X
4X
VSS
VDD
32X
24X
16X
8X
60
ONE
S32X
S24X
S16X
S8X
y_{6i+2}
y_{6i+5}
72
74
76
78
80
82
84
70

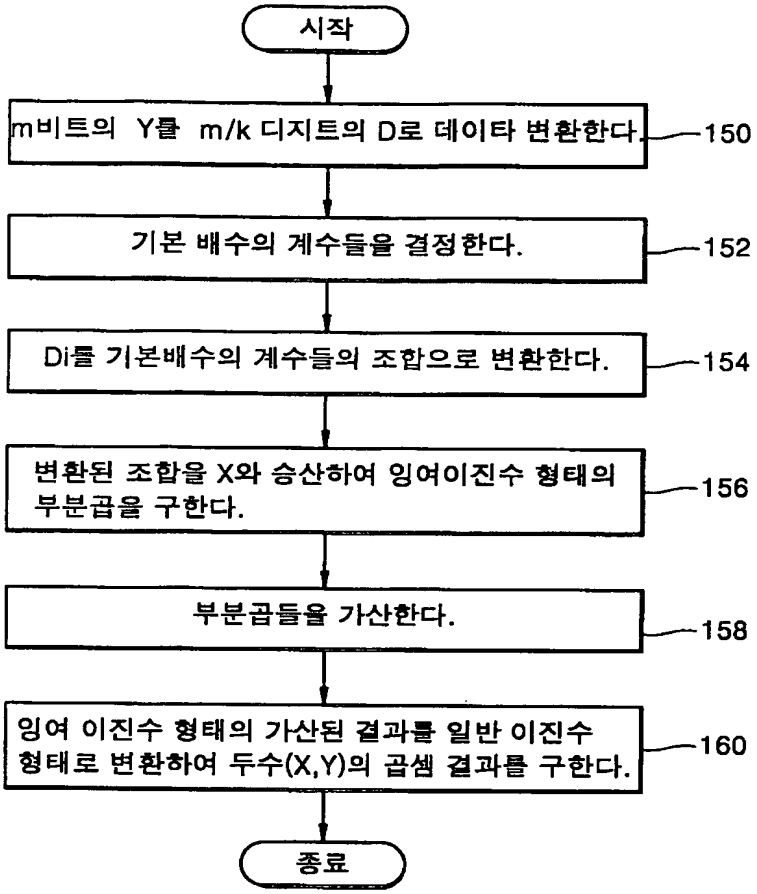
【도 4】


92
94
98
100
90
96
$\overline{(b_i^+ \oplus b_i^-)}$
$\overline{l_i}$
h_{i-1}
$\overline{\alpha_i}$
β_i
β_{i-1}
$d_i^+\overline{d_i^-}$
$\overline{(d_i^+ \oplus d_i^-)}$
$b_i^+\overline{b_i^-}$
h_i

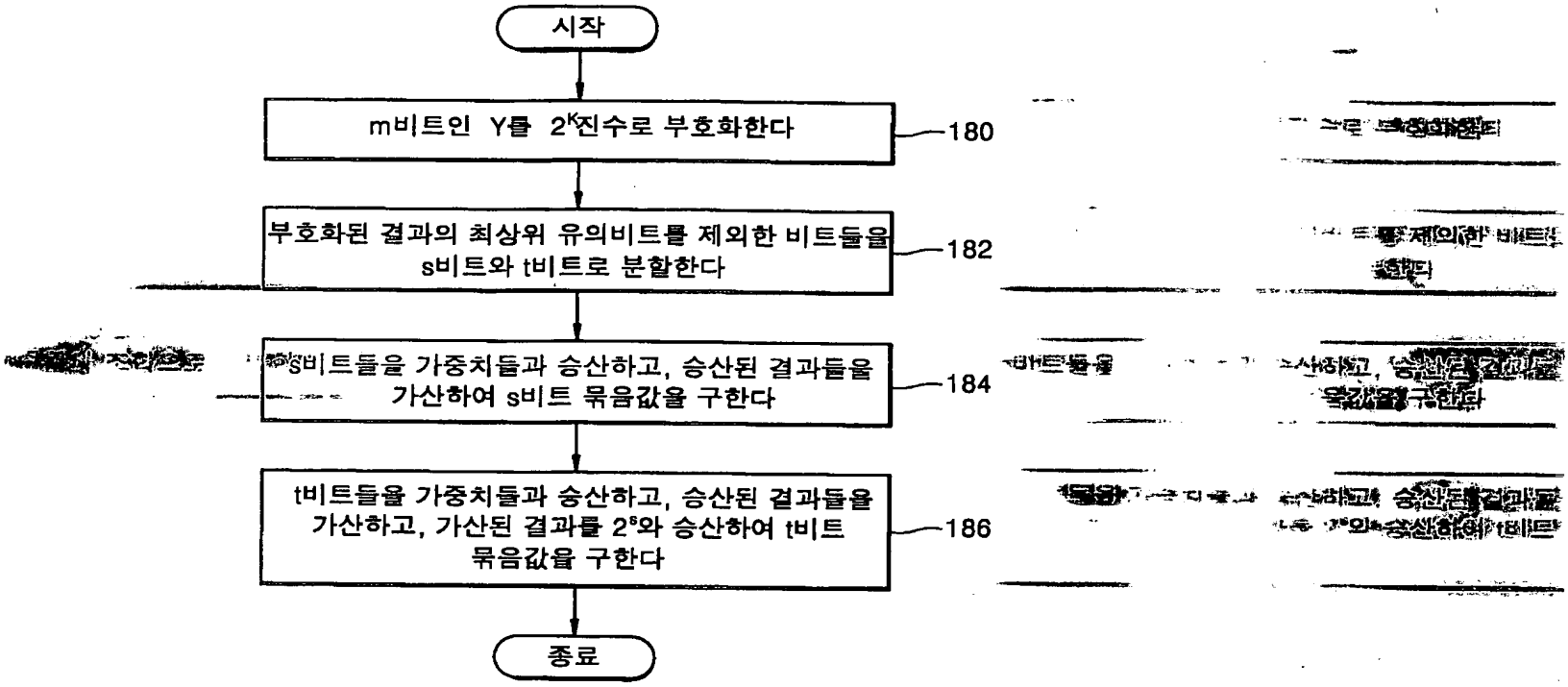
【도 5】


$\overline{(a_i^+ \oplus a_i^-)}$
116
118
$\overline{(b_i^+ \oplus b_i^-)}$
108
l_i
120
122
124
126
128
130
132
134
136
$\overline{\alpha_i}$
$d_i^+\overline{d_i^-}$
$\overline{(d_i^+ \oplus d_i^-)}$
$\overline{h_{i-1}}$
$\overline{(a_i^+ \oplus a_i^-)}$
114
$\overline{k_i}$
β_i β_{i-1}
$a_i^+\overline{a_i^-}$
$b_i^+\overline{b_i^-}$
110
$\overline{h_i}$
112

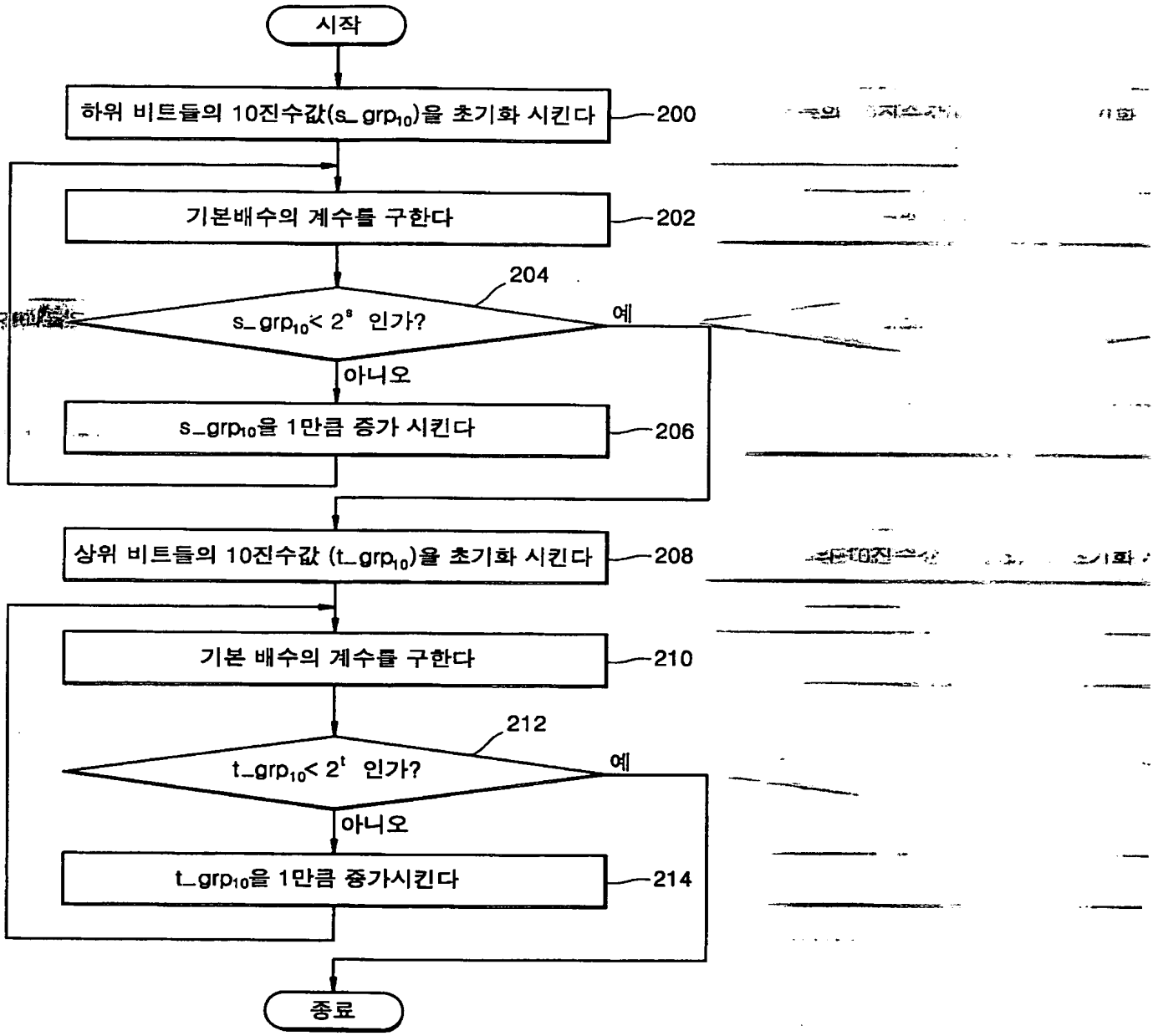
【도 6】


시작
m비트의 Y를 m/k 디지트의 D로 데이타 변환한다.
150
기본 배수의 계수들을 결정한다.
152
Di를 기본배수의 계수들의 조합으로 변환한다.
154
변환된 조합을 X와 승산하여 잉여이진수 형태의 부분곱을 구한다.
156
부분곱들을 가산한다.
158
잉여 이진수 형태의 가산된 결과를 일반 이진수 형태로 변환하여 두수(X,Y)의 곱셈 결과를 구한다.
160
종료

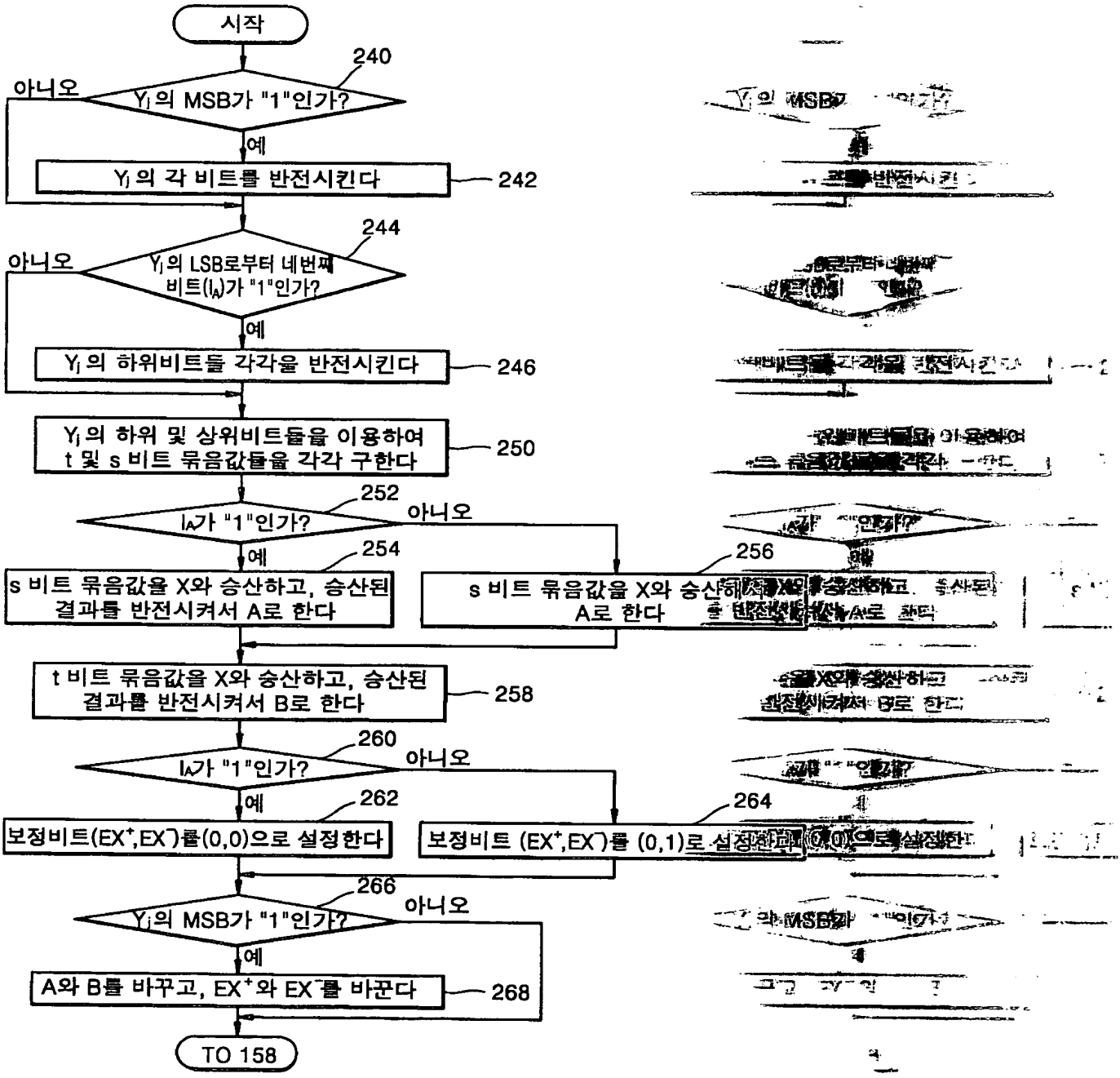
【도 7】


시작
m비트인 Y를 2^K진수로 부호화한다
180
부호화된 결과의 최상위 유의비트를 제외한 비트들을 s비트와 t비트로 분할한다
182
s비트들을 가중치들과 승산하고, 승산된 결과들을 가산하여 s비트 묶음값을 구한다
184
t비트들을 가중치들과 승산하고, 승산된 결과들을 가산하고, 가산된 결과를 2^s와 승산하여 t비트 묶음값을 구한다
186
종료

【도 8】


시작
하위 비트들의 10진수값(s_grp_{10})을 초기화 시킨다
200
기본배수의 계수를 구한다
202
204
$s\_grp_{10} < 2^s$ 인가?
예
아니오
s_grp_{10}을 1만큼 증가 시킨다
206
상위 비트들의 10진수값 (t_grp_{10})을 초기화 시킨다
208
기본 배수의 계수를 구한다
210
212
$t\_grp_{10} < 2^t$ 인가?
예
아니오
t_grp_{10}을 1만큼 증가시킨다
214
종료

【도 9】


시작
240
아니오
Y_j의 MSB가 "1"인가?
예
Y_j의 각 비트를 반전시킨다
242
244
아니오
Y_j의 LSB로부터 네번째 비트(I_A)가 "1"인가?
예
Y_j의 하위비트들 각각을 반전시킨다
246
Y_j의 하위 및 상위비트들을 이용하여 t 및 s 비트 묶음값들을 각각 구한다
250
252
I_A가 "1"인가?
아니오
예
254
s 비트 묶음값을 X와 승산하고, 승산된 결과를 반전시켜서 A로 한다
256
s 비트 묶음값을 X와 승산하고, 승산된 결과를 A로 한다
t 비트 묶음값을 X와 승산하고, 승산된 결과를 반전시켜서 B로 한다
258
260
I_A가 "1"인가?
아니오
예
262
보정비트(EX^+,EX^-)를(0,0)으로 설정한다
264
보정비트 (EX^+,EX^-)를 (0,1)로 설정한다
266
Y_j의 MSB가 "1"인가?
아니오
예
A와 B를 바꾸고, EX^+와 EX^-를 바꾼다
268
TO 158